Algorithmen und Programmierung 3, WS 2003/2004 — 10. Übungsblatt

Abgabe bis Donnerstag, 29. Januar 2004

57. (0 Punkte) Eine *Multimenge* (engl. *multiset* oder *bag*) ist etwas Ähnliches wie eine Menge, außer dass Elemente auch mehrfach vorkommen dürfen. Die Reihenfolge spielt keine Rolle. Zum Beispiel ist $\{\!\{a, b\}\!\} \neq \{\!\{a, a, b\}\!\} = \{\!\{a, b, a\}\!\} \neq \{\!\{a, a, a, b\}\!\}$, für $a \neq b$.
    (a) Schreiben Sie eine Spezifikation für einen abstrakten Datentyp von Multimengen über der Grundmenge der ganzen Zahlen (`int`), die folgende Operationen unterstützt: Erzeugen einer leeren Multimenge; Einfügen und Streichen eines Elementes (dabei wird die Vielfachheit jeweils um 1 erhöht beziehungsweise erniedrigt); Feststellen der Vielfachkeit eines Elementes.
    (b) Geben Sie eine konkrete Darstellung (etwa als Java-Klasse `Multimenge`) an. Beschreiben Sie die Abstraktionsfunktion, sowie die Invarianten, die die gültigen Darstellungen charakterisieren. Geben Sie auch die Vorbedingungen für alle Operatiomen an. (Sie dürfen dabei vernünftige Einschränkungen für die verfügbaren Operationen machen.)
    (c) Implementieren Sie die Operationen. Sie können von der in der Vorlesung besprochenen Implementierung für Mengen mit bis zu 100 Elementen[1] ausgehen.
    (d) Beweisen Sie die Korrektheit Ihrer Implementierung.

58. (11 Punkte) Betrachten wir die Knoten v eines Graphen in der Reihenfolge, wie der rekursive Aufruf $T(v)$ bei der Tiefensuche *beendet* wird. Das heißt, wir fügen *am Ende* des Programms $T(v)$ folgende Zeile ein:

    ```
    num2++; T2Nummer[v] := num2;
    ```

    (a) (4 Punkte) Beweisen Sie: Wenn es eine Kante (u, v) mit $\mathrm{T2Nummer}[v] > \mathrm{T2Nummer}[u]$ gibt, dann enthält der Graph einen Kreis.
    (b) (4 Punkte) Wie kann man diesen Kreis bestimmen? Schreiben Sie ein Programmstück für diese Aufgabe.
    (c) (3 Punkte) Verwenden Sie die Tatsache aus Aufgabe (a), um aus der T2Nummerierung eines kreisfreien Graphen eine topologische Sortierung zu berechnen, sofern bei der Tiefensuche alle Knoten besucht werden. Beschreiben Sie den Algorithmus in Worten.

59. (0 Punkte) Bei der Tiefensuche werden möglicherweise nicht alle Knoten des Graphen besucht. Was muss man tun, damit das Verfahren der vorigen Aufgabe immer funktioniert?

60. (0 Punkte) Zeigen Sie, wie man Tiefensuche ohne Rekursion sehr einfach mit einem Stapel für noch zu bearbeitende *Kanten* implementieren kann.

61. (9 Punkte) Betrachten Sie das folgende einfache Haskell-Programm:

    ```
    camba [ ] _ = True
    camba _ [ ] = False
    camba (x:xs) (y:ys) = (x==y) && camba xs ys
    klungo [ ] _ = True
    klungo _ [ ] = False
    klungo xs (y:ys) = camba xs (y:ys) || klungo xs ys
    ```

    (a) (1 Punkt) Beschreiben Sie in Worten, was diese beiden Funktionen berechnen.
    (b) (4 Punkte) Spezifizieren Sie die beiden Funktionen mathematisch (modellierend).
    (c) (4 Punkte) Wie viele Vergleiche der Form `x==y` werden bei den folgenden Eingaben durchgeführt?

    ```
    klungo "aaaab" "aaaaaaaaaa"
    klungo "abababc" "bababababa"
    ```

[1] `http://www.inf.fu-berlin.de/~rote/Lere/2003-04-WS/Algorithmen+Programmierung3/Menge.java`