Prof. Dr. Uwe Schmidt

fhwedel
UNIVERSITY OF APPLIED SCIENCES

4. Februar 2014

Aufgaben zur Klausur **Softwaredesign** im WS 2013/14 (BInf v310, BMInf v300, BWInf v310, BInf 23, BMInf 23, BWInf 23)

Zeit: 90 Minuten erlaubte Hilfsmittel: keine

Bitte tragen Sie Ihre Antworten und fertigen Lösungen ausschließlich an den gekennzeichneten Stellen in das Aufgabenblatt ein. Ist ihre Lösung wesentlich umfangreicher, so überprüfen Sie bitte nochmals Ihren Lösungsweg.

Nutzen Sie die Rückseiten der Klausur zur Entwicklung der Lösungen und übertragen die fertigen Lösungen in das Aufgabenblatt.

Sollten Unklarheiten oder Mehrdeutigkeiten bei der Aufgabenstellung auftreten, so notieren Sie bitte, wie Sie die Aufgabe interpretiert haben.

Viel Erfolg!

Diese Klausur besteht einschließlich dieses Deckblattes aus 9 Seiten.

Aufgabe 1:

Ein Präfix-Baum ist eine Datenstruktur zur effizienten Implementierung von Maps, bei denen die Schlüssel aus Zeichenreihen bestehen.

Ein Präfixbaum besteht aus einem (Wurzel-)Knoten. Dieser enthält möglicherweise eine Information, einen Attributwert, für einen Schlüssel. Außerdem besteht er aus einer Menge von Nachfolgern. Jeder Nachfolger wird durch ein Zeichen identifiziert und ist wieder ein Präfixbaum.

Das 1. Zeichen eines Schlüssels wird zur Markierung der Kanten auf der 1. Ebene des Baumes verwendet, das 2. Zeichen auf der 2. Ebene usw. Der Schlüssel für einen Eintrag setzt sich also zusammen aus den Zeichen des Pfades von der Wurzel bis zu dem Knoten, der das Attribut enthält.

Tipp: Veranschaulichen Sie sich diese Datenstruktur, indem Sie z.B. für die Menge der Schlüssel-Wert-Paare $\{(a, 5), (anna, 9), (ei, 7), (eto, 9)\}$ so einen Baum skizzieren.

Entwickeln Sie einen Datentyp in Haskell-Notation für einen Typ, der diese Datenstruktur 1-1 (ohne Optimierungen und Datenstrukturverfeinerungen) beschreibt. Die Attributkomponente soll dabei generisch sein.

. . . . . .

. . . . . .

. . . . . .

Gibt es für alle Mengen von Schlüssel-Wert-Paaren einen zugehörigen Präfixbaum?

ja ☐ nein ☐

Begründung:

. . . . . .

. . . . . .

Gibt es für alle Mengen von Schlüssel-Wert-Paaren **genau** einen zugehörigen Präfixbaum?

ja ☐ nein ☐

Begründung:

. . . . . .

. . . . . .

Aufgabe 2:

Ein gerichteter Graph, bei dem die Knoten durch einen Wertebereich *NodeId* eindeutig identifiziert werden und sowohl an den Knoten Zusatzinformation vom Typ *NodeAttr* als auch an den Kanten Zusatzinformation der Art *EdgeAttr* enthalten ist, kann durch folgende Datentypen modelliert werden.

.0 type $Graph$ $= \mathsf{Map}\ NodeId\ \ NodeInfo$

.1 type $NodeInfo$ $= (NodeAttr, Succ)$

.2 type $Succ$ $= \mathsf{Map}\ NodeId\ \ EdgeAttr$

Transformieren Sie dieses Modell in ein neues, das geeignet ist, 1–1 in Datenbankschemata für ein relationales Datenbanksystem umgesetzt zu werden. Das Modell muss also in eine Menge von Relationstypen umgesetzt werden, deren Elementbereiche unstrukturiert sind oder aus Tupeln von unstrukturierten Bereichen besteht.

Nehmen Sie an, dass *NodeId* ein unstrukturierter Datentyp ist und *NodeAttr* und *EdgeAttr* folgendermaßen definiert sind, wobei die A_i ebenfalls alle unstrukturiert sind.

.0 type $NodeAttr$ $= (A_1, [\,A_2\,])$

.1 type $EdgeAttr$ $= (A_3, A_4)$

Das transformierte Datenmodell in Haskell-Notation

1) .....

2) .....

3) .....

4) .....

5) .....

6) .....

7) .....

8) .....

9) .....

10) .....

Aufgabe 3:

Gegeben seien die folgende Datentypen:

.0	data $Stmt$	$= Assignment\ Variable\ Expr$
.1		$\mid IfStmt\ Expr\ Stmt\ Stmt$
.2		$\mid WhileStmt\ Expr\ Stmt$
.3		$\mid Sequence\ [Stmt]$
.4	type $Variable$	$= \ldots$
.5	type $Expr$	$= \ldots$

Entwickeln Sie für den Datentyp *Stmt* gleichwertige Klassendefinitionen in Java.

Aufgabe 4:

Gegeben sei ein Relationenschema $R = (\{A_1, A_2, A_3, A_4\}, \{A_1, A_2\})$ für die Definition einer Tabelle in einer relationalen Datenbank. Die A_i sind die Attribute des Relationenschemas R, A_1 und A_2 bilden den Primärschlüssel. Hier wird angenommen, dass alle Attributwerte Zeichenreihen (Strings) sind. Entwickeln sie ein gleichwertiges Datenmodell in abstrakter Syntax in Haskell-Notation.

. . . . . .

. . . . . .

. . . . . .

Gegeben sei ein zweites Relationenschema $R_2 = (\{A_5, A_6, A_7\}, \{A_5, A_6, A_7\})$. Transformieren Sie dieses ebenfalls in eine abstrakte Syntax.

. . . . . .

. . . . . .

. . . . . .

Aufgabe 5:

Bei dem Erzeugen von Objekten durch einen Prototyp spielt das Kopieren (Klonen) von Objekten eine wichtige Rolle.

Gegeben sei das folgende Java–Programmfragment:

```
public class PrototypeFactory {
    public X makeX() {
        return pt.copy();
    }
    private final
        X pt = new X(new Y(new Z(3),4),5);
}
class X {
    Y d1;
    int d2;
    X(Y d1, int d2) {
        this.d1 = d1;
        this.d2 = d2;
    }
    X copy() {
        return ...;
    }
    ...
}
class Y {
    Z d1;
    int d2;
    Y(Z d1, int d2) {
        this.d1 = d1;
        this.d2 = d2;
    }
    Y copy() {
        return ...;
    }
    ...
}
class Z {
    int d2;
    Z(int d2) {
        this.d2 = d2;
    }
    Z copy() {
        return ...;
    }
    ...
}
```

Wie viele neue Objekte müssen beim Kopieren des Prototypen in einem Aufruf von *makeX* erzeugt werden, wenn ...

1. ... die Variablen d1 und d2 in X über Methoden verändert werden können, für alle anderen Variablen in den übrigen Klassen dieses aber nicht möglich ist.

   . . . . . .

2. ... die Variablen d1 und d2 in Y über Methoden verändert werden können, für alle anderen Variablen in den übrigen Klassen dieses aber nicht möglich ist.

   . . . . . .

3. ... keine der Variablen in den Klassen X, Y und Z über Methoden verändert werden können.

   . . . . . .

4. ... die Variable d2 in Z über Methoden verändert werden kann, für alle anderen Variablen in den übrigen Klassen dieses aber nicht möglich ist.

   . . . . . .

5. ... nur die Variable d1 in X über Methoden verändert werden kann, für alle anderen Variablen in den Klassen X, Y und Z dieses aber nicht möglich ist.

   . . . . . .

Aufgabe 6:

Gegeben sei das folgende Datenmodell in abstrakter Syntax in Haskell–Notation:

```
.0  data Table    = Leaf String Attr
.1               | Entry Table Attr
.2               | Switch CMap
.3               | Single Char Table
.4               | Empty
.5  type CMap     = Map Char Table
.6  type String   = [Char]
.7  type Attr     = Set Int
```

Welche Strukturmuster kommen in diesem Modell vor?
Geben Sie jeweils den Musternamen und die beteiligten Datentypen und Kostruktornamen an.

1) .....

2) .....

3) .....

4) .....

5) .....

6) .....

7) .....

8) .....
