Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Frühjahr

2005

66112

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

- Prüfungsaufgaben -

Fach: **Informatik (vertieft studiert)**

Einzelprüfung: **Automatentheorie, Komplexität, Algorithmen**

Anzahl der gestellten Themen (Aufgaben): 2

Anzahl der Druckseiten dieser Vorlage: 9

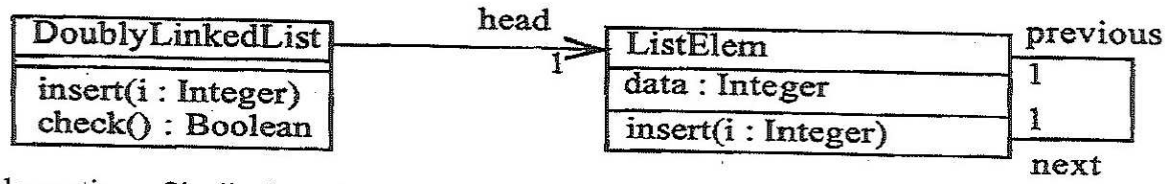
Bitte wenden!

Thema Nr. 1

Sämtliche Teilaufgaben sind zu bearbeiten!

Aufgabe 1

Betrachten Sie folgendes Klassendiagramm, das doppelt-verkettete Listen spezifiziert. Die Assoziation `head` zeigt auf das erste Element der Liste. Die Assoziationen `previous` und `next` zeigen auf das vorherige bzw. folgende Element.



Implementieren Sie die doppelt-verketteten Listen in einer geeigneten objektorientierten Sprache (z. B. Java oder C++), das heißt:

a) Implementieren Sie die Klasse `ListElem`. Die Methode `insert` ordnet eine ganze Zahl i in eine aufsteigend geordnete doppelt-verkettete Liste l an die korrekte Stelle ein. Sei z. B. das Objekt l eine Repräsentation der Liste <0, 2, 2, 6, 8>; dann liefert `l.insert (3)` eine Repräsentation der Liste <0, 2, 2, 3, 6, 8>.

b) Implementieren Sie die Klasse `DoublyLinkedList`, wobei die Methode `insert` eine Zahl i in eine aufsteigend geordnete Liste einordnet. Die Methode `check` überprüft, ob eine Liste korrekt verkettet ist, d. h. ob für jedes ListElem-Objekt `o`, das über den `head` der Liste erreichbar ist, der Vorgänger des Nachfolgers von `o` gleich `o` ist.

Aufgabe 2

a) Was ist eine reguläre Sprache und was ist eine kontextfreie Sprache? Definieren Sie die beiden Begriffe und erklären Sie, worin sich die beiden Sprachklassen unterscheiden.
Formulieren Sie das Pumpinglemma für reguläre Sprachen.

b) Gegeben sei die Sprache

$$L = \{w \in \{a, b\}^* \mid |w|_a = |w|_b, \text{ und für jedes Präfix p von w gilt } |p|_b \leq |p|_a\}.$$

Dabei bezeichnet $|w|_x$ für einen Buchstaben x die Anzahl der Vorkommen von x in w.

b1) Welche der folgenden Worte sind in der Sprache L enthalten: abab, baba, aaabba, aababbab? Begründen Sie kurz Ihre Antwort.

b2) Zeigen Sie, dass L nicht regulär ist.

b3) Zeigen Sie, dass die Sprache L kontextfrei ist, indem Sie einen Kellerautomaten $K = (Z, \Sigma, \Gamma, \delta, z, \#, \varnothing)$ finden, der L mit leerem Keller akzeptiert, und erläutern Sie kurz die Funktionsweise von K. Geben Sie eine Folge von Konfigurationen an, die K beim Akzeptieren des Wortes abaabbab durchläuft.

Fortsetzung nächste Seite!

Aufgabe 3

a) Erläutern Sie den Begriff der dynamischen Bindung bei Methodenaufrufen in objekt-orientierten Sprachen.

b) Die folgenden Java-Klassen sollen Daten über Personen speichern. Die Methode `getSalary` in der Klasse `Person` soll das Gehalt einer Person als Ergebnis liefern. Die Methode `getSalary` in der Klasse `Manager` soll die Summe aus dem Managerzuschlag und dem Personen-Gehalt als Ergebnis liefern.

```
class Person {
   String name;
   double salary;
   public double getSalary() {
      return salary;
   }
}
class Manager extends Person {
   double extraSalary;
   public double getSalary() {
      return (extraSalary + getSalary());
   }
   public static void main(String[] args) {
      Person p = new Manager();
      p.salary = 5000;
      p.getSalary();
   }
}
```

Terminiert die Methode `main`? Begründen Sie Ihre Antwort. Wie kann man die Implementation verbessern?

c) Erklären Sie die Parameterübergabemechanismen *call by value* und *call by reference.*

d) Welche Ausgabewerte liefert die folgende Methode `main`? Erklären Sie insbesondere die letzte Ausgabezeile. Begründen Sie Ihre Antwort.

```
class CallBy {
   static int changei(int i) {
      return (++i);
   }
   public static void main (String[] args) {
      int i = 2 ;

      System.out.println("i ist vorher "+i);
      int i_return = changei(i);
      System.out.println ("changei lieferte "+i_return+" zurueck");
      System.out.println ("i ist nachher "+i) ;
   }
}
```

Fortsetzung nächste Seite!

Aufgabe 4

a) Erläutern Sie die auf Floyd und Hoare zurückgehende Verifikationsmethode. (In Ihrer Antwort müssen Sie mindestens die Begriffe von Zusicherung, schwächste Vor- und stärkste Nachbedingung erklären.)

b) Eine Bank bietet ihren Kunden 1% Zinsen pro Monat. Betrachten Sie das folgende Hoare-Triple (*), das das Anwachsen eines Geldbetrags B beschreibt, wenn ein Kunde diesen eine gewisse Anzahl von Monaten M auf seinem Konto stehen lässt.

(*)
```
{betrag = B, Zeitraum = M, M > 0}
while (Zeitraum > 0) {
betrag = betrag*(1 + 1/100);
Zeitraum = Zeitraum - 1;
}
{betrag=B*(1+1/100)^M}
```

b1) Zeigen Sie, dass $\{betrag = B \cdot (1+ 1/100)^{M\text{-}zeitraum}, zeitraum \geq 0\}$ eine Schleifeninvariante ist.
b2) Beweisen Sie die Gültigkeit von (*).
b3) Terminiert das Programm immer? Beweisen Sie Ihre Antwort.

Thema Nr. 2

Sämtliche Teilaufgaben sind zu bearbeiten!

Aufgabe 1

Betrachten Sie die beiden regulären Ausdrücke über dem Alphabet $\Sigma = \{a, b\}$:

$$\alpha = ab(aab)^*, \quad \beta = (aba)^* ab.$$

a) Zeigen Sie die Äquivalenz der beiden Ausdrücke, d.h. $\mathcal{L}(\alpha) = \mathcal{L}(\beta)$.
b) Geben Sie eine Grammatik an, die $\mathcal{L}(a)$ erzeugt.
c) Geben Sie einen deterministischen endlichen Automaten an, der $\mathcal{L}(a)$ akzeptiert.

Aufgabe 2

Es seien $\Sigma = \{a, b\}$ und $\Gamma = \{a, b, c\}$ Alphabete. Betrachten Sie die Sprachen

$$L = \{w \in \Sigma^*; |w|_{ab} = |w|_{ba}\} \quad \text{und} \quad L' = \{w \in \Gamma^*; |w|_{ab} = |w|_{ba}\}$$

aller Wörter, in denen das Teilwort ab genauso oft vorkommt, wie das Teilwort ba. (Beachten Sie: als Wortmengen sind L und L' identisch, aber es wird auf unterschiedliche Alphabete Bezug genommen!)

a) Zeigen Sie, dass die Sprache $L \subseteq \Sigma^*$ regulär ist, indem Sie sowohl einen regulären Ausdruck als auch einen akzeptierenden endlichen Automaten für L angeben.
b) Zeigen Sie, dass die Sprache $L' \subseteq \Gamma^*$ nicht regulär ist, indem Sie (beispielsweise) nachweisen, dass L' unendlichen Index $\mathcal{I}(L')$ in Γ^* hat.
c) Zeigen Sie, dass die Sprache $L' \subseteq \Gamma^*$ kontextfrei ist, indem Sie einen Kellerautomaten über Γ konstruieren, der L' akzeptiert.

Aufgabe 3

In dieser Aufgabe bezeichne μ den Operator der *Minimalisierung*, der einer $(k+1)$-stelligen (partiellen) Funktion $f : \mathbb{N}^{k+1} \to \mathbb{N}$ eine k-stellige (partielle) Funktion $\mu(f) : \mathbb{N}^k \to \mathbb{N}$ zuordnet.

a) Geben Sie die Definition von $\mu(f)$ an, wobei Sie der genauen Beschreibung des Definitionsbereiches besondere Beachtung schenken sollten. Insbesondere: welche unterschiedlichen Gründe können dafür verantwortlich sein, dass $\mu(f)$ für ein $(x_1, \ldots, x_k) \in \mathbb{N}^k$ nicht definiert ist?

b) Es sei nun f die zweistellige (partielle) Funktion

$$f(x,y) = \begin{cases} |x-y| & \text{falls } x \neq 3 \text{ und } y \neq 3 \\ y+3 & \text{falls } x = 3 \\ \uparrow & \text{falls } y = 3 \text{ und } x \neq 3 \end{cases}$$

Dabei steht $\uparrow$ für „undefiniert". Berechnen Sie $\mu(f)$.

Fortsetzung nächste Seite!

Aufgabe 4

Gegeben seien die Methoden foo(n) und bar(n) mit asymptotischen Aufwandsfunktionen O(log n) bzw. O(n). Es seien:

- k eine positive Konstante,
- n eine Variable,
- i und j zwei Laufvariablen.

Geben Sie den asymptotischen Aufwand (Komplexität) folgender Programmstücke in O-Notation an:

a)
```
for   (int i = 1; i <= n; i++){
      foo(i);
}
bar(n);
```

b)
```
for   (int i = 1; i <= k; i++){
      foo(n);
      bar(n);
}
```

c)
```
for   (int i = 1; i <= n; i++){
      bar(i);
      for (int j = 1; j <= k; j++){
            foo(n);
      }
}
```

d)
```
for   (int i = 1 ; i <= n; i++){
      for (int j = 1; j <= i; j++){
           bar(j));
      }
}
```

Aufgabe 5

Folgendes Programmstück realisiert ein Sortieren durch Einfügen

```
1   for (int j = 1; j < feld.length; j++) {
2       schluessel = feld[j];
3       // Fuege Element j in sortierte Folge
4       // feld[0]..feld[j-1] ein
5       int i = j - 1;
6       while (i >= 0 && feld[i] > schluessel) {
7           feld[i + 1] = feld[i];
8           i = i - 1;
9       }
10      feld[i + 1] = schluessel;
11  }
```

Fortsetzung nächste Seite!

a) Geben Sie für die Programmzeilen 1, 2, 6 und 7 an, wie oft diese Zeilen im besten sowie im schlechtesten Fall ausgeführt werden. Nehmen Sie an, dass das Feld 10 Elemente enthält und alle Variablen korrekt vereinbart wurden!

b) Geben Sie eine Aufwandsabschätzung in O-Notation für das Verhalten im schlechtesten Fall an.

c) Im obigen Code wird die Einfügestelle mit linearer Suche gefunden. Wieso verschlechtert sich durch die Verwendung der Binärsuche der Aufwand im günstigsten Fall? Erläutern Sie Ihre Antwort gegebenenfalls an einer Zahlenfolge.

Aufgabe 6

Die Zahl π soll nach folgender Formel berechnet werden: *Hinweis: Wenn Sie die Zahl π auf eine andere Weise berechnen, z. B. durch einen Zugriff auf die Java-Bibliothek, ist Ihre Lösung ungültig.*

$$\pi = (4/1 - 4/3 + 4/5 - 4/7 ...)$$

Gegeben sei folgender Rahmen

```
1   static final double EPSILON = 0.0001;
2
3   public static double piBerechnung(char schalter) {
4       double pi = 0.0;
5       switch (schalter) {
6       case 'r' :
7           pi = piBerechnungRekursiv(1.0, 1.0);
8           break;
9       case 'i' :
10          pi = piBerechnungIterativ();
11          break;
12      default:
13          throw new RuntimeException("Fehlende Methode");
14      } //switch
15      return pi;
16  } // piBerechnung
```

Die Berechnung soll abbrechen, wenn der Wert des Bruchs unter den vorgegebenen Wert `EPSILON` gefallen ist. Geben Sie bei jeder Ihrer Lösungen an, ob in Ihrer Lösung der erste berechnete Bruch, der unter dem Wert von `EPSILON` liegt, noch zur Summe mit hinzuaddiert wird oder nicht. Beide Varianten sind erlaubt.

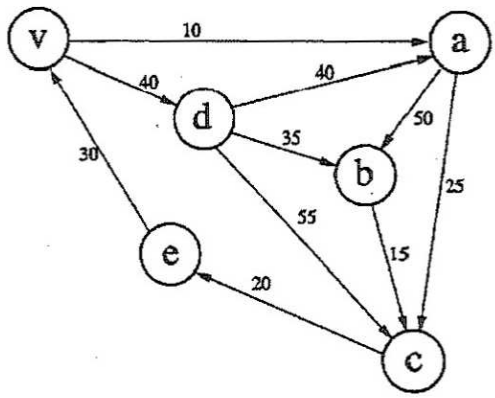
a) Geben Sie eine rekursive Methode `piBerechnungRekursiv` an, die π entsprechend der oben angegebenen Formel berechnet und über eine Schnittstelle wie in der Methode `piBerechnung` angegeben verfügt.
Verwenden Sie den ersten Parameter zur Übergabe des aktuell betrachteten Nenners. Der zweite Parameter soll zur Implementierung des Vorzeichens dienen.

Fortsetzung nächste Seite!

b) Geben Sie eine iterative Methode `piBerechnungIterativ` an, die π entsprechend der oben angegebenen Formel berechnet und über eine Schnittstelle wie in der Methode `piBerechnung` angegeben verfügt.

Aufgabe 7

Berechnen Sie mit dem Algorithmus von Dijkstra die kürzesten Pfade für den Knoten v.



a) Machen Sie den Ablauf des Algorithmus mit Hilfe einer Tabelle folgenden Musters deutlich:

Schritt	ausgewählter Knoten	untersuchter Knoten	Knotenbewertung nach Dijkstras Algorithmus					
./.	./.	./.	v	a	b	c	d	e
0	-	-	0	∞	∞	∞	∞	∞
1	v							

b) Geben Sie die kürzesten Pfade ausgehend von Knoten v zu jedem einzelnen der übrigen Knoten an!

Fortsetzung nächste Seite!

Aufgabe 8

Gegeben seien die folgenden Zahlen: 7, 4, 3, 5, 0, 1

a) Zeichnen Sie eine Hash-Tabelle mit 8 Zellen und tragen Sie diese Zahlen genau in der oben gegebenen Reihenfolge in Ihre Hash-Tabelle ein. Verwenden Sie dabei die Streufunktion $f(n) = n^2 \ mod \ 7$ und eine Kollisionsauflösung durch lineares Sondieren.

b) Welcher Belegungsfaktor ist für die Streutabelle und die Streufunktion aus Teilaufgabe a zu erwarten, wenn sehr viele Zahlen eingeordnet werden und eine Kollisionsauflösung durch Verkettung (verzeigerte Listen) verwendet wird? Begründen Sie Ihre Antwort kurz.

Hinweis: Es ist kein formaler Beweis nötig, aber Sie müssen Ihre Antwort plausibel begründen.

Aufgabe 9

Gegeben sei folgende partielle Spezifikation eines Datentyps Baum mit der Eigenschaft, dass genau eine Integer-Zahl pro Knoten gespeichert wird.

Signatur:

neu:		$\rightarrow$ *Baum*
konstr:	*Baum* $\times$ *Baum* $\times$ *int*	$\rightarrow$ *Baum*
links:	*Baum*	$\rightarrow$ *Baum*
rechts:	*Baum*	$\rightarrow$ *Baum*
anzahl:	*Baum*	$\rightarrow$ *int*

Axiome:

links(*konstr*(*b*1,*b*2,*k*))	=	*b*1
rechts(*konstr*(*b*1,*b*2,*k*))	=	*b*2

a) Die Funktion `anzahl` soll angeben, wieviel `int-Werte` im Baum enthalten sind. Geben Sie Axiome an, die diese Funktion festlegen.

b) Eine Klasse enthalte bereits die den Axiomen entsprechenden Methoden `links` und `rechts` mit folgenden Signaturen:

- *Baum links(Baum b)*
- *Baum rechts(Baum b)*

Geben Sie den Rumpf der folgenden Methode an:
public int anzahl (Baum b)

c) Geben Sie an, wie die Reihenfolge bezeichnet wird, in der die Knoten in Ihrer Lösung der Methode `anzahl` durchlaufen werden.