

Klausur Informatik I

20.02.2003

Prof. Dr. G. Goos
Dipl.-Inform. M. L. Noga

Vorname:

Name:

Zur Klausur sind keine Hilfsmittel und kein eigenes Papier zugelassen. Die Bearbeitungszeit beträgt 60 Minuten.

Die Klausur ist komplett und geheftet abzugeben. **Nur Blätter, die Namen und Matrikelnummer tragen, gehen in die Bewertung ein.**

Sie dürfen die Rückseite der Aufgabenblätter als Konzeptpapier benutzen. **Nur Lösungen, die sich auf dem entsprechenden Aufgabenblatt oder seiner Rückseite befinden, gehen in die Bewertung ein.**

Aufgabe	1	2	3	4	5	6	Σ
Maximal	10	10	9	10	11	10	60
K1							
K2							
K3							

Punkte:

Note :

Aufgabe 1: Wissen (10 Punkte)

Welche der folgenden Aussagen sind wahr (*W*) und welche falsch (*F*)? Kreuzen Sie an. Jedes korrekte Kreuz zählt 0,5 Punkte, jedes falsche Kreuz kostet 0,5 Punkte Abzug. Diese Aufgabe wird mindestens mit 0 Punkten bewertet.

1. Es gibt zwei Darstellungen der Null bei Verwendung

W *F* von Vorzeichen und Betrag

W *F* des Einerkomplements für negative Zahlen

W *F* des Zweierkomplements für negative Zahlen

2. Es gibt ein kleinstes Element in jeder

W *F* halbgeordneten Menge

W *F* vollständig halbgeordneten Menge

W *F* total geordneten Menge

3. *W* *F* Jeder Baum ist schwach zusammenhängend.

4. *W* *F* Jeder schwach zusammenhängende Graph ist ein Baum.

5. *W* *F* Jeder bipartite Graph ist unzusammenhängend.

6. *W* *F* Jedes Petrinetz hat eine konstante Anzahl Stellen.

7. *W* *F* Jedes lebendige Petrinetz hat eine monoton wachsende Anzahl Marken.

8. *W* *F* $\lambda x.(\lambda y.z)$ ist ein Churchscher Wahrheitswert.

9. *W* *F* Der λ -Kalkül arbeitet mit α -, β - und γ -Konversionen.

10. Eine Ordnung auf den Elementen ist Voraussetzung für den ADT

W *F* Schlange

W *F* Prioritätsschlange

W *F* Binärer Suchbaum

11. Ohne Verteilungsannahme enthält die Aufwandsklasse $O(n \log(n))$

W *F* Sortieren durch Einfügen (Insertion sort)

W *F* Sortieren durch Mischen (Mergesort)

W *F* Sortieren durch Zerlegen (Quicksort)

12. *W* *F* Binäre Suche auf Listen ist $O(\log(n))$.

	K1	K2	K3
A1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Aufgabe 2: Formale Sprachen (2+3+5=10 Punkte)

Sei $G_i = (\Sigma, N, P_i, S)$ mit $\Sigma = \{a, b, c, d\}$, $N = \{A, B, C, D, S\}$ und

$$\begin{array}{lll}
 P_1 = \{S \rightarrow A, & P_2 = \{S \rightarrow ABC, & P_3 = \{S \rightarrow aA, \\
 A \rightarrow BC, & AB \rightarrow a, & A \rightarrow bB \mid cC, \\
 B \rightarrow bBb \mid D, & BC \rightarrow a, & B \rightarrow aA, \\
 C \rightarrow cCc \mid D, & aC \rightarrow aaCa \mid c, & C \rightarrow aA \mid dD, \\
 D \rightarrow d & Aa \rightarrow aAaa \mid c \} & D \rightarrow d & \}
 \end{array}$$

Geben Sie jeweils den engsten Chomsky-Typ der Grammatik, die von der Grammatik erzeugte Sprache sowie den engsten Chomsky-Typ dieser Sprache an.

Jedes korrekte Kreuz zählt 0,5 Punkte, jedes falsche Kreuz kostet 0,5 Punkte Abzug. Diese Aufgabe wird mindestens mit 0 Punkten bewertet.

G_1 liegt in CH - 0 CH - 1 CH - 2 CH - 3

$L(G_1) =$

$L(G_1)$ liegt in CH - 0 CH - 1 CH - 2 CH - 3

G_2 liegt in CH - 0 CH - 1 CH - 2 CH - 3

$L(G_2) =$

$L(G_2)$ liegt in CH - 0 CH - 1 CH - 2 CH - 3

G_3 liegt in CH - 0 CH - 1 CH - 2 CH - 3

$L(G_3) =$

$L(G_3)$ liegt in CH - 0 CH - 1 CH - 2 CH - 3

	K1	K2	K3
A2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Aufgabe 3: SQL (3+3+3=9 Punkte)

Die städtische Bibliothek Boxingen speichert ihre Stammdaten in folgenden Tabellen:

Exemplare

Feld	Typ
exNr	int
buchNr	int
einkauf	date

Bücher

Feld	Typ
buchNr	int
titel	String
autor	String

Kunden

Feld	Typ
kundenNr	int
name	String
adresse	String

Ausleihen

Feld	Typ
kundenNr	int
exNr	int
faellig	date

a) Mahngebühren sind die wichtigste Einnahmequelle der Bibliothek. Formulieren Sie eine SQL-Anfrage, welche die Namen aller Benutzer mit überfälligen Ausleihen ermittelt.

b) Ein Kunde möchte ein Buch mit gegebener Buchnummer entleihen. Formulieren Sie eine SQL-Anfrage, welche die Nummern der nicht verliehenen Exemplare dieses Buches ermittelt.

c) Unlängst wurde ein Buch der Bibliothek in beschädigtem Zustand aufgefunden. Die Buchnummer war unkenntlich, allein die Exemplarnummer konnte abgelesen werden. Formulieren Sie eine SQL-Anfrage, welche die Exemplarnummern aller Exemplare dieses Buches ermittelt.

A3

K1

K2

K3

Aufgabe 4: Formale Logik (6+4=10 Punkte)

a) Sei \mathcal{B} eine boolesche Algebra. Zeigen Sie: Gilt für zwei gegebene $a, b \in \mathcal{B}$ die Identität $a \wedge b = a$, so gilt für sie auch $a \vee b = b$. (Hinweis: Beginnen Sie Ihre Umformungen bei b).

b) Bringen Sie die aussagenlogische Formel $F = \neg((x \rightarrow y) \equiv z)$ in konjunktive Normalform.

Aufgabe 5: Haskell (3+4+4=11 Punkte)

a) Sie müssen dringend eine Zeichenkette umkehren ("hallo" \rightarrow "ollah"), haben aber versehentlich die Standardbibliothek gelöscht. Nur Mustererkennung und die Listenoperation `:` funktionieren noch. Implementieren Sie mit diesen Mitteln die Funktion `reverse :: [a] -> [a]`.

b) Gegeben sei eine duplikatfreie Liste `l :: [a]`. Wir bezeichnen die Liste aller möglichen Teillisten mit Auslassungen als Kombinationen `c :: [[a]]` von `l`. Beispiel: die Kombinationen von `[1,2,3]` sind `[[1,2,3], [1,2], [1,3], [1], [2,3], [2], [3], []]`. Die relative Position jeder einzelnen Kombination ist dabei unerheblich.

Schreiben Sie eine Funktion `combs :: [a] -> [[a]]`, welche die Kombinationen ihres Arguments berechnet. Sie können beliebige Funktionen der Standardbibliothek verwenden.

c) Vereinbaren Sie einen Datentyp `BSTree k v` für binäre Suchbäume. Knoten, die nicht dem leeren Baum entsprechen, tragen Schlüssel vom Typ `k` und Werte vom Typ `v`. Definieren Sie eine Funktion `contains :: Ord k => (BSTree k v) -> k -> Bool`, die prüft, ob ein binärer Suchbaum einen gegebenen Schlüssel enthält.

Aufgabe 6: Asymptotischer Aufwand (3+2+2+3=10 Punkte)

Gegeben sei folgende Funktion:

```
func xs k | k < len = func smaller k
          | k == len = pivot
          | k > len = func (tail larger) (k-len-1)
where pivot  = head xs
      smaller = filter (< pivot) xs
      larger  = filter (>=pivot) xs
      len     = length smaller
```

a) Geben Sie die Signatur der Funktion an. Beschreiben Sie in einem Satz, was die Funktion tut.

b) Betrachten Sie zunächst den lokalen Aufwand eines Aufrufes von `func` ohne rekursiven Abstieg. Definieren Sie ein Maß der Problemgröße n und geben Sie den asymptotischen Aufwand folgender Teilberechnungen an (Operationen auf Listenelementen haben konstanten Aufwand):

```
 $n :=$ 

pivot

smaller,
larger

len
```

c) Nehmen Sie an, die Anordnung der Elemente von `xs` sei rein zufällig. Geben Sie die Rekurrenzgleichung für den erwarteten Aufwand f von `func` an.

d) Lösen Sie die Rekurrenz. Welchen asymptotischen erwarteten Aufwand hat `func`?

A6

K1

K2

K3