Musterloesung

Klausur Informatik 1
13. Oktober 2001

Jähnichen / Flik
Godehardt / Neunast

Name: ....

Vorname: ....

Matr.-Nr.: ....

Bearbeitungszeit: 120 Minuten

- Benutzen Sie für die Lösung der Aufgaben nur das mit diesem Deckblatt ausgeteilte Papier. **Lösungen, die auf anderem Papier geschrieben werden, können nicht gewertet werden!**
- Schreiben Sie Ihre Lösungen auf das Aufgabenblatt der jeweiligen Aufgabe. Verwenden Sie auch die Rückseiten. **Schreiben Sie keine Lösungen einer Aufgabe auf ein Blatt, das nicht zu dieser Aufgabe gehört!** Wenn Sie zusätzliche, von uns ausgegebene Blätter verwenden, geben Sie unbedingt an, zu welcher Aufgabe die Lösung gehört!
- Schreiben Sie deutlich! Doppelte, unleserliche oder mehrdeutige Lösungen können nicht gewertet werden!
- Schreiben Sie nur in **blau** oder **schwarz**. Lösungen, die mit Bleistift geschrieben sind, können nicht gewertet werden!
- Bei der Implementation von OPAL-Funktionen brauchen *keine* Importe mit angegeben zu werden, wenn Funktionen aus dem gegebenen Teil der Bibliotheca Opalica (oder aus der Struktur `Nat`, `Real`, `Seq` oder den Strukturen für die Listenfunktionale) oder sonstigen (in den Aufgaben) gegebenen Strukturen benutzt werden!
- Die OPAL-Implementierungen müssen nicht dokumentiert werden.
- Erscheint Ihnen eine Aufgabe mehrdeutig, wenden Sie sich an die Betreuer.
- Tragen Sie jetzt (vor Beginn der Bearbeitungszeit) auf *allen* Blättern Ihren Namen und Ihre Matrikelnummer ein.

Aufgabe	Punkte	erreicht
1	8	
2	14	
3	11	
4	1	
5	6	
6	4	
7	5	

Musterloesung

1. Aufgabe (8 Punkte): Datentypen

Gegeben sei der im Folgenden definierte Datentyp `person`, mit dessen Hilfe sich der Datentyp `stammbaum` definieren lässt. Der Stammbaum enthält die Daten einer Person sowie deren Abstammung auf Seiten der Mutter (`ms`) und auf Seiten des Vaters (`vs`), also die Daten der Mutter bzw. des Vaters.

```
DATA person == person (name: denotation, vorname: denotation, alter: nat)

DATA stammbaum == unbekannt
                  stammbaum (person : person, ms : stammbaum, vs : stammbaum)
```

1.1. Induzierte Signatur (3 Punkte) Geben Sie zu dem oben definierten Datentyp `stammbaum` die induzierte Signatur an. Wie nennt man die einzelnen induzierten Elemente?

Lösung:

```
-- Sorte
SORT stammbaum

-- Konstruktoren
FUN stammbaum : denotation ** denotation ** nat -> stammbaum
FUN unbekannt : stammbaum

-- Selektoren
FUN person : stammbaum -> person
FUN ms : stammbaum -> stammbaum
FUN vs : stammbaum -> stammbaum

-- Diskriminatoren
FUN stammbaum? : stammbaum -> bool
FUN unbekannt? : stammbaum -> bool
```

Um welche Art von Datentypen handelt es sich bei `person` und `stammbaum`?

Lösung:

person $\Longrightarrow$ *Produkttyp*
stammbaum $\Longrightarrow$ *Summentyp*

1.2. Baumtraversierung (5 Punkte) Benennen Sie drei mögliche Traversierungsarten für einen Datentypen Baum. Deklarieren und definieren Sie eine Funktion `personen`, die alle Personen eines gegebenen Stammbaumes als Liste liefert, indem Sie eine der Traversierungsarten verwenden. Geben Sie an, welche Art der Traversierung Sie verwendet haben.

Lösung:

Preorder:

```
FUN personen : stammbaum -> seq[person]
DEF personen(unbekannt) == <>
DEF personen(stammbaum(p, ms, vs)) ==
                      p :: (personen(ms) ++ personen(vs))
```

Inorder:

```
FUN personen : stammbaum -> seq[person]
DEF personen(unbekannt) == <>
DEF personen(stammbaum(p, ms, vs)) ==
                      personen(ms) ++ (p :: personen(vs))
```

Postorder:

```
FUN personen : stammbaum -> seq[person]
DEF personen(unbekannt) == <>
DEF personen(stammbaum(p, ms, vs)) ==
                      (personen(ms) ++ personen(vs)) +% p
```

Musterloesung

2. Aufgabe (14 Punkte): Funktionen höherer Ordnung

Fixpunktberechnung. Eine Zahl x heißt *Fixpunkt* einer Funktion f, falls die Gleichung $f(x) = x$ erfüllt ist. Für manche Funktionen f können wir einen Fixpunkt bestimmen, indem wir mit einem Anfangsschätzwert beginnen und f wiederholt auf diesen Wert anwenden, bis das Ergebnis sich nur noch geringfügig ändert.

Es entsteht bei der Berechnung also die Kette:

$$f(x), f(f(x)), f(f(f(x))), \ldots$$

2.1. Fixpunkt (6 Punkte) Deklarieren und definieren Sie eine Funktion `fixpunkt`, die als Argumente eine einstellige reellwertige Funktion und einen Anfangsschätzwert bekommt und eine Näherungslösung für einen Fixpunkt der Funktion als reelle Zahl berechnet.

Lösung:

```
FUN fixpunkt : (real -> real) ** real -> real
DEF fixpunkt (f, schaetzwert) ==
    LET naechstes == f(schaetzwert)
    IN
        IF gut?(schaetzwert, naechstes)
           THEN naechstes
           ELSE fixpunkt(f, naechstes)
        FI
```

Hilfsfunktionen:

```
FUN toleranz : real
DEF toleranz == "0.00001"!

FUN gut? : real ** real -> bool
DEF gut? (v1, v2) ==
    abs(v1 - v2) < toleranz
```

Hinweis: Überlegen Sie sich eine geeignete Möglichkeit zur Berechnung der ungefähren Gleichheit zweier reeller Zahlen.

Mittelwertdämpfung. Leider konvergiert dieser Algorithmus zur Fixpunktberechnung nicht für alle Funktionen f. Es besteht jedoch die Möglichkeit, statt des Funktionswertes $f(x)$ den Mittelwert von x und $f(x)$ als neuen Schätzwert zu verwenden, um dem abzuhelfen. Dieses als Mittelwertdämpfung bezeichnete Prinzip führt darüber hinaus zur schnelleren Konvergenz einiger Funktionen f als das herkömmliche Verfahren der Fixpunktberechnung.

2.2. Mittelwertberechnung (4 Punkte) Deklarieren und definieren Sie eine Funktion `mittelwert`, die eine Funktion f als Argument bekommt und eine Funktion als Ergebnis liefert, die den Mittelwert zwischen x und $f(x)$ berechnet.

Lösung:

```
FUN mittelwert : (real -> real) -> (real -> real)
DEF mittelwert (f)(x) ==
    (f(x) + x) / 2
```

2.3. Mittelwertdämpfung (4 Punkte) Benutzen Sie die Funktionen `fixpunkt` und `mittelwert`, um eine Funktion `daempfung` zu deklarieren und definieren, die den Fixpunkt einer gegebenen Funktion mit Hilfe der Mittelwertdämpfung berechnet. Der Anfangsschätzwert soll ebenfalls Argument der Funktion `daemfung` sein.

Lösung:

```
FUN daempfung : (real -> real) ** real -> real
DEF daempfung (f, x) ==
    fixpunkt( mittelwert(f), x)
```

Musterloesung

3. Aufgabe (11 Punkte): Sequenzen

In dieser Aufgabe gilt es, in einer gegebenen Sequenz von natürlichen Zahlen die längste Teilsequenz zu finden, die aufsteigend sortiert ist.

3.1. Aufteilen einer Sequenz (5 Punkte) Deklarieren und definieren Sie eine Funktion `teilen`, die eine gegebene Sequenz in zwei Teilsequenzen aufteilt, wobei die erste Teilsequenz alle vom Sequenzanfang an aufsteigenden Elemente und die zweite Teilsequenz den Rest der Sequenz enthält.

Zum Beispiel würde die Sequenz $< 1, 9, 1, 2, 3, 5, 7, 2, 3, 7 >$ aufgeteilt werden in die Sequenzen $< 1, 9 >$ und $< 1, 2, 3, 5, 7, 2, 3, 7 >$.

Lösung:

```
FUN teilen :  seq[nat] -> seq[nat] ** seq[nat]
DEF teilen (<>) == (<>, <>)
DEF teilen (x :: <>) == (x :: <>, <>)
DEF teilen (x :: (y :: r)) ==
    IF x <= y
       THEN LET (s1, s2) == teilen(y :: r)
               IN
            (x :: s1, s2)
       ELSE
            (x :: <>, y :: r)
    FI
```

3.2. Längste Teilsequenz (6 Punkte) Mit Hilfe der Funktion `teilen` ist es möglich, die längste Teilsequenz zu ermitteln, die aufsteigend sortiert ist. Deklarieren und definieren Sie eine Funktion `laengste`, die zu einer gegebenen Sequenz die längste aufsteigend sortierte Teilsequenz ermittelt.

Zum Beispiel ist die längste aufsteigend sortierte Teilsequenz der Sequenz $< 1, 9, 1, 2, 3, 5, 7, 2, 3, 7 >$ die Sequenz $< 1, 2, 3, 5, 7 >$.

Lösung:

```
FUN laengste : seq[nat] -> seq[nat]
DEF laengste (s) == laengste(<>, s)

FUN laengste :  seq[nat] ** seq[nat] -> seq[nat]
DEF laengste (bisherige, <>) == bisherige
DEF laengste (bisherige, rs) ==
    LET (neue, rest) == teilen(rs)
       IN
    IF #(neue) > #(bisherige)
       THEN laengste(neue, rest)
       ELSE laengste(bisherige, rest)
    FI
```

Name: ....

Matr.-Nr. ....

Musterloesung

4. Aufgabe (1 Punkte): Codes

Gegeben ist ein mit Kreuzsicherung (gerade Längs- und gerade Querparität) gesicherter Datenblock. Markieren Sie fehlerhafte Bits und interpretieren Sie den korrigierten Datenblock als ASCII-Zeichenfolge.

1	1	0	0	1	0	0	1
1	1	1	0	1	1	1	0
0	1	1	0	0	1	1	0
0	1	1	0	1	1	1	1
1	0	1	0	0	0	0	1
1	0	0	1	1	1	1	1

Lösung:

1	*1*	*0*	*0*	*1*	*0*	*0*	*1*	*'I'*
1	*1*	*1*	*0*	*1*	*1*	*1*	*0*	*'n'*
0	*1*	*1*	*0*	*0*	*1*	*1*	*0*	*'f'*
0	*1*	*1*	*0*	*1*	*1*	*1*	*1*	*'o'*
1	*0*	*1*	<u>*1*</u>	*0*	*0*	*0*	*1*	*'1'*
1	*0*	*0*	*1*	*1*	*1*	*1*	*1*	*BCC*

				b_8	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	
				b_7	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	H
				b_6	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	E
				b_5	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	X
b_4	b_3	b_2	b_1		00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	
0	0	0	0	00			SP	0	@	P	‘	p			NS	°	À	Ð	à	ð	0
0	0	0	1	01			!	1	A	Q	a	q			¡	±	Á	Ñ	á	ñ	1
0	0	1	0	02			"	2	B	R	b	r			¢	²	Â	Ò	â	ò	2
0	0	1	1	03			#	3	C	S	c	s			£	³	Ã	Ó	ã	ó	3
0	1	0	0	04			$	4	D	T	d	t			¤	´	Ä	Ô	ä	ô	4
0	1	0	1	05			%	5	E	U	e	u			¥	µ	Å	Õ	å	õ	5
0	1	1	0	06			&	6	F	V	f	v			¦	¶	Æ	Ö	æ	ö	6
0	1	1	1	07			#	7	G	W	g	w			§	·	Ç	×	ç	÷	7
1	0	0	0	08			(	8	H	X	h	x			¨	¸	È	Ø	è	ø	8
1	0	0	1	09			)	9	I	Y	i	y			©	¹	É	Ù	é	ù	9
1	0	1	0	10			*	:	J	Z	j	z			ª	º	Ê	Ú	ê	ú	A
1	0	1	1	11			+	;	K	[	k	{			«	»	Ë	Û	ë	û	B
1	1	0	0	12			,	<	L	\	l	|			¬	¼	Ì	Ü	ì	ü	C
1	1	0	1	13			-	=	M	]	m	}			SH	½	Í	Ý	í	ý	D
1	1	1	0	14			.	>	N	^	n	~			®	¾	Î	Þ	î	þ	E
1	1	1	1	15			/	?	O	_	o	DE			¯	¿	Ï	ß	ï	ÿ	F
	H	E	X		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

5. Aufgabe (6 Punkte): Zahlendarstellungen

Musterloesung

5.1. (1 Punkte) Stellen Sie 110010101101_2 zur Basis 8 und 16 dar.

Lösung:

$110010101101_2 = 6255_8 = CAD_{16}$

5.2. (1 Punkte) Wandeln Sie 469_{10} in eine vorzeichenlose Dualzahl um. Die Rechenschritte müssen erkennbar sein!

Lösung:

$$
\begin{array}{rcl}
469/2 = & 234 & \textit{Rest } 1 \\
234/2 = & 117 & \textit{Rest } 0 \\
117/2 = & 58 & \textit{Rest } 1 \\
58/2 = & 29 & \textit{Rest } 0 \\
29/2 = & 14 & \textit{Rest } 1 \\
14/2 = & 7 & \textit{Rest } 0 \\
7/2 = & 3 & \textit{Rest } 1 \\
3/2 = & 1 & \textit{Rest } 1 \\
1/2 = & 0 & \textit{Rest } 1 \\
469_{10} = & 111010101_2 &
\end{array}
$$

5.3. (2 Punkte) Addieren Sie im Dualsystem 0110_2 und 0111_2 und geben Sie die resultierenden Bedingungsbits an. Notieren Sie die auftretenden Überträge.

$$
\begin{array}{ccccc}
 & 0 & 1 & 1 & 0 \\
+ & 0 & 1 & 1 & 1 \\
\hline
 & & & & \\
\hline\hline
\end{array}
$$

Lösung:

$$
\begin{array}{ccccc}
 & 0 & 1 & 1 & 0 \\
+ & 0 & 1 & 1 & 1 \\
 & \bullet & \bullet & & \\
\hline
 & 1 & 1 & 0 & 1 \\
\hline\hline
\end{array}
\quad c=0,\ z=0,\ n=1,\ v=1
$$

5.4. (1 Punkte) Subtrahieren Sie im Dualsystem 1010_2 und 0111_2. Notieren Sie die auftretenden Überträge.

$$
\begin{array}{ccccc}
 & 1 & 0 & 1 & 0 \\
- & 0 & 1 & 1 & 1 \\
\hline
 & & & & \\
\hline\hline
\end{array}
$$

Lösung:

$$
\begin{array}{ccccc}
 & 1 & 0 & 1 & 0 \\
- & 0 & 1 & 1 & 1 \\
 & \bullet & \bullet & \bullet & \\
\hline
 & 0 & 0 & 1 & 1 \\
\hline\hline
\end{array}
$$

5.5. (1 Punkte) Wandeln Sie $+0.00101001_2$ in das aus der Vorlesung bekannte Gleitpunktformat (1-Bit-Vorzeichen, 4-Bit-Exponent, 11-Bit-Mantisse, Bias = 7) um.

Musterloesung

Lösung:

Normalisieren:	$0.00101001_2 = 1.01001_2 \cdot 2^{-3}$
Exponent:	$-3 + 7 = 4 = 0100_2$
Ergebnis:	*0 0100 01001000000*

6. Aufgabe (4 Punkte): Boolesche Algebra

Musterloesung

6.1. (2 Punkte) Vergleichen Sie die Ausdrücke $\overline{a}+\overline{b}+\overline{c}$ und $ab \rightarrow \overline{c}$ mit Hilfe der folgenden Tabelle.

a	b	c	$\overline{a}+\overline{b}$	$\overline{a}+\overline{b}+\overline{c}$	ab	$ab \rightarrow \overline{c}$

Lösung:

a	b	c	$\overline{a}+\overline{b}$	$\overline{a}+\overline{b}+\overline{c}$	ab	$ab \rightarrow \overline{c}$
0	0	0	1	1	0	1
0	0	1	1	1	0	1
0	1	0	1	1	0	1
0	1	1	1	1	0	1
1	0	0	1	1	0	1
1	0	1	1	1	0	1
1	1	0	0	1	1	1
1	1	1	0	0	1	0

$$\overline{a}+\overline{b}+\overline{c} = ab \rightarrow \overline{c}$$

6.2. (1 Punkte) Formen Sie den folgenden Ausdruck in eine disjunktive Normalform um.

$\overline{a} \oplus (b+c) =$

Lösung:

$$\begin{aligned}\overline{a} \oplus (b+c) &= a \cdot (b+c) + \overline{a} \cdot \overline{(b+c)} \\ &= a \cdot b + a \cdot c + \overline{a} \cdot \overline{b} \cdot \overline{c}\end{aligned}$$

6.3. (1 Punkte) Formen Sie den folgenden Ausdruck so um, daß ausschließlich NOT- und NOR-Funktionen verwendet werden.

$(\overline{a}+b) \cdot c =$

Musterloesung

Lösung:

$$
\begin{aligned}
(\overline{a}+b)\cdot c &= \overline{\overline{(\overline{a}+b)\cdot c}} \\
&= \overline{\overline{\overline{a}+b}+\overline{c}}
\end{aligned}
$$

Musterloesung

7. Aufgabe (5 Punkte): Schaltnetzentwurf

Die Funktion f für $x \in \{0, \ldots, 15\}$ ist definiert durch

$$y = f(x) := \begin{cases} 1 & \text{wenn} \quad x \in \{1, 5, 8, 9\} \\ 0 & \text{sonst} \end{cases}$$

Zu entwerfen ist ein Schaltnetz, wobei die 16 Argumente als vierstellige Dualzahlen dargestellt werden sollen.

7.1. (1 Punkte) Füllen Sie dazu die Tabelle aus.

x_3	x_2	x_1	x_0	y

Lösung:

x_3	x_2	x_1	x_0	y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

7.2. (3 Punkte) Ermitteln Sie die booleschen Gleichungen für y in minimaler disjunktiver Normalform sowie in minimaler konjunktiver Normalform.

Name: ....

Matr.-Nr. ....

A7

Musterloesung

y: (x_0 over columns 2–3, x_1 over columns 3–4; x_2 beside rows 2–3, x_3 beside rows 3–4)

$\overline{y}$: (x_0 over columns 2–3, x_1 over columns 3–4; x_2 beside rows 2–3, x_3 beside rows 3–4)

Lösung:

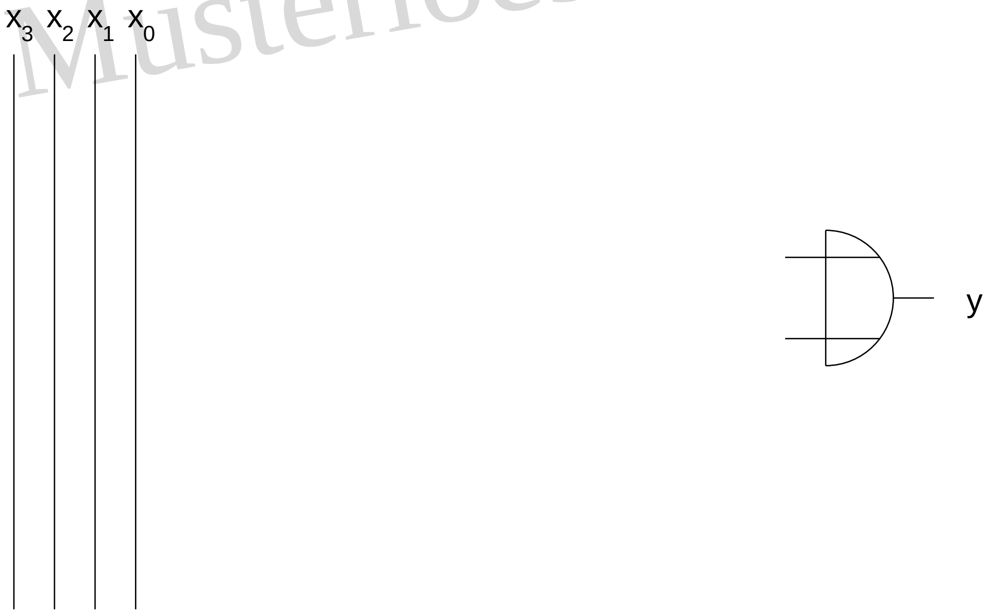
y: (x_0 over columns 2–3, x_1 over columns 3–4; x_2 beside rows 2–3, x_3 beside rows 3–4)

0	1	0	0
0	1	0	0
0	0	0	0
1	1	0	0

$$
\begin{aligned}
y &= \overline{x_3}\cdot\overline{x_1}\cdot x_0 + x_3\cdot\overline{x_2}\cdot\overline{x_1} \\
&= \overline{\overline{x_3}\cdot\overline{x_0} + x_1 + x_2\cdot x_3} \\
&= (x_3 + x_0)\cdot x_1\cdot(\overline{x_2}+\overline{x_3})
\end{aligned}
$$

Musterloesung

7.3. (1 Punkte) Vervollständigen Sie das Blockbild für y in disjunktiver Normalform.



Lösung: