Technische Universität München
Fakultät für Informatik
Prof. Dr. M. Broy

WS 2001/2002
11. April 2002

Lösungsvorschläge der Klausur zu Einführung in die Informatik I

A

Aufgabe 1 Bindung, Gültigkeit und Lebensdauer (Lösungsvorschlag)

Bindungen	Programm	Lebensdauer							Gültigkeit						
		n	n	n	m	even	isodd	iseven	n	n	n	m	even	isodd	iseven
	fct even = (**nat** n) **bool**:	*				*			*				*		
	⌈ **var nat** m := n;	*			*	*	*	*	*			*	*	*	*
	fct iseven = (**nat** n) **bool**:	*	*		*	*	*	*		*		*	*	*	*
	if n = 0	*	*		*	*	*	*		*		*	*	*	*
	then true	*	*		*	*	*	*		*		*	*	*	*
	else ⌈ m := n-1;	*	*		*	*	*	*		*		*	*	*	*
	isodd() ⌋	*	*		*	*	*	*		*		*	*	*	*
	fi;	*	*		*	*	*	*		*		*	*	*	*
	fct isodd = ( ) **bool**:	*			*	*	*	*	*			*	*	*	*
	if m = 0	*			*	*	*	*	*			*	*	*	*
	then false	*			*	*	*	*	*			*	*	*	*
	else ⌈ **var nat** n := m-1;	*		*	*	*	*	*			*	*	*	*	*
	iseven(n) ⌋	*		*	*	*	*	*			*	*	*	*	*
	fi;	*			*	*	*	*	*			*	*	*	*
	iseven(m) ⌋	*			*	*	*	*	*			*	*	*	*

Aufgabe 2 Termersetzung (Lösungsvorschlag)

a) Das Termersetzungssystem der Funktion `if_then_else` lautet:

```
if_then_else  True   b  c  →  b  (1.5 Punkte)
if_then_else  False  b  c  →  c  (1.5 Punkte)
```

b) Das Termersetzungssystem der Funktion `equal` lautet (3 Punkte):

```
equal  a  b  →
 if_then_else (a, if_then_else(b, True, False), if_then_else(b, False, True))
```

Aufgabe 3 Boolesche Algebra (Lösungsvorschlag)

a) **1.5 Punkte**
g(3, 0, 7) = g(3, 1, 1*7) = g(3, 2, 2*7) = g(3, 3, 3*14) = 42

b) **2 Punkte**
Abstiegsfunktion: $h : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{N};\ h(x,y,z) = \begin{cases} 0 & falls\, x \leq y \\ x-y & falls\, x > y \end{cases}$

c) **ca. 4.5 Punkte**
Terminierungsbeweis zur Funktion g mit Hilfsfunktion h:
Wir zeigen für alle $k \in \mathbb{N}$: $\forall (x,y,z) \in \mathbb{N}^3 : h(x,y,z) \leq k \Rightarrow g(x,y,z) \neq \perp$.

Beweis durch vollständige Induktion über k:
Indulktionsanfang $k = 0$:
Sei $(x,y,z) \in \mathbb{N}^3$ mit $h(x,y,z) \leq 0$. Dann gilt nach Definition von h: $x \leq y$ und somit nach Definition von g: $g(x,y,z) = 0 \neq \perp$.

Induktionsschritt $k \to k+1$:
Sei $(x,y,z) \in \mathbb{N}^3$ mit $h(x,y,z) \leq k+1$. (O.b.d.A sei $h(x,y,z) \neq 0$, also $x > y$). Dann gilt:

$$k+1 \geq h(x,y,z) = x-y > x-y-1 = x-(y+1) = h(x,y+1,(y+1)*z).$$

Also gilt $k \geq h(x,y+1,(y+1)*z)$. Dann folgt mit der Indunktionsvoraussetzung $g(x,y+1,(y+1)*z) \neq \perp$ und aus der Definition von g folgt:

$$g(x,y,z) = g(x,y+1,(y+1)*z) \neq \perp.$$

□

Aufgabe 4 Zuweisungsorientierte Programmierung (Lösungsvorschlag)

a) Eine Implementierung der zuweisungsorientierten Prozedur `test` hat in Vorlesungsnotation folgende Gestalt:

```
proc test = (nat i, nat j):
    if (i ≥ 2 ∧ i ≤ j ) then
                    ⌈ var nat n := i;
                      while (n ≤ j) do
                                ⌈ primfak (n);
                                  n := n + 1; ⌋
                                od ⌋
    fi
```

b) Eine zuweisungsorientierte Lösung der Funktion `kleinsterTeiler` lautet:

```
fct kleinsterTeiler = (nat n) nat:
  ⌈ var bool b:= false ;
    var nat d := 2;
    while (¬ b) do
              ⌈ if (¬(mod(n, d) =?= 0)) then d := d+1
                                       else b := true
                fi ⌋
    od;
    d ⌋
```

c) Eine Implementierung von `primfak` in Vorlesungsnotation lautet:

```
proc primfak = (nat n):
    ⌈ if (n≥2) then
            ⌈ var nat m := n;
              var nat d := kleinsterTeiler(m);
              writeNat (d) ;
              while (d<m) do
                        ⌈ m := m/d;
                          d := kleinsterTeiler(m);
                          writeNat(d); ⌋
                        od ⌋
      fi ⌋
```

Aufgabe 5 Fixpunkttheorie (Lösungsvorschlag)

a) Die gesuchte Rechenvorschrift lautet:

```
fct f = (nat x) bool:
   if   x =? 0 then true
   elif x =? 1 then false
               else f(x-2)
   fi
```

b) Berechnung von f_1:

$$f_1(x) = \tau[f_0](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ f_0(x-2) & x > 1 \\ \perp & x = \perp \end{cases}$$

Einsetzen von $f_0(x)$ führt auf:

$$f_1(x) = \tau[f_0](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ \perp & (x = \perp) \lor (x > 1) \end{cases}$$

Berechnung von f_2:

$$f_2(x) = \tau[f_1](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ f_1(x-2) & x > 1 \\ \perp & x = \perp \end{cases}$$

Einsetzen von f_1 ergibt:

$$f_2(x) = \tau[f_1](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ \texttt{true} & (x-2) = 0 \\ \texttt{false} & (x-2) = 1 \\ \perp & ((x-2) > 1) \lor ((x-2) = \perp) \\ \perp & x = \perp \end{cases}$$

und somit:

$$f_2(x) = \tau[f_1](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ \texttt{true} & x = 2 \\ \texttt{false} & x = 3 \\ \perp & (x > 3) \lor (x = \perp) \end{cases}$$

c) Zu zeigen ist, dass die Funktionalgleichung $f_\infty(x) = \tau[f_\infty](x)$ für $f_\infty(x) = ((x \bmod 2) \stackrel{?}{=} 0)$, wobei $x \in \mathbb{N}$, erfüllt ist. Wir betrachten $\tau[f_\infty](x)$ für $x \in \mathbb{N}$; dann folgt:

$$\tau[f_\infty](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ f_\infty(x-2) & x > 1 \end{cases}$$

Einsetzen von f_∞ ergibt:

$$\tau[f_\infty](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ (((x-2) \bmod 2) \stackrel{?}{=} 0) & x > 1 \end{cases}$$

Nun gilt jedoch $((x-2) \bmod 2) = (x \bmod 2)$, woraus folgt:

$$\tau[f_\infty](x) = \begin{cases} \texttt{true} & x = 0 \\ \texttt{false} & x = 1 \\ ((x \bmod 2) \stackrel{?}{=} 0) & x > 1 \end{cases}$$

Da außerdem $((0 \bmod 2) \stackrel{?}{=} 0) = \texttt{true}$ und $((1 \bmod 2) \stackrel{?}{=} 0) = \texttt{false}$ ist, erhalten wir:

$$\tau[f_\infty](x) = \begin{cases} ((x \bmod 2) \stackrel{?}{=} 0) & x = 0 \\ ((x \bmod 2) \stackrel{?}{=} 0) & x = 1 \\ ((x \bmod 2) \stackrel{?}{=} 0) & x > 1 \end{cases}$$

und schließlich:

$$\tau[f_\infty](x) = f_\infty(x)$$

□