Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

14. August 1998

Aufgaben zur Klausur **Grundlagen der Programmierung** im SS 98 (WI v003, II v003)

Zeit: 120 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.

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:

Berechnen Sie die disjunktive Form des Ausdrucks

$(a \Rightarrow (\neg b \vee \neg c)) \wedge (\neg a \Rightarrow (b \wedge c))$

Die disjunktive Form erlaubt Negation nur vor Variablen, Variablen und negierte Variablen dürfen mit $\wedge$ verknüpft werden, die so geformten Ausdrücke dürfen mit $\vee$ verknüpft werden.

Ergebnis:

.....

.....

.....

Aufgabe 2:

Gegeben sei eine Variable f für ein Feld

var

f : array $[0..n-1]$ of Z

und die folgenden prädikatenlogischen Formeln

1. $\forall\, 0 < i < n-1 \bullet f[i] \geq 0 \Rightarrow (f[i-1] < 0 \Leftrightarrow f[i+1] \geq 0)$
2. $\forall\, 0 < i < n-1 \bullet f[i] \geq 0 \wedge f[i+1] \geq 0 \wedge f[i-1] \geq 0$
3. $\forall\, 0 < i < n-1 \bullet f[i] \geq 0 \Rightarrow (f[i-1] < 0 \oplus f[i+1] < 0)$
4. $\forall\, 0 \leq i < n-2 \bullet f[i] \geq 0 \vee f[i+1] \geq 0 \vee f[i+2] \geq 0$
5. $\forall\, 0 < i < n-1 \bullet f[i] \geq 0 \Rightarrow (f[i-1] < 0 \vee f[i+1] < 0)$
6. $\forall\, 0 < i < n-1 \bullet f[i] \geq 0 \vee f[i+1] \geq 0 \vee f[i-1] \geq 0$
7. $\forall\, 0 < i < n-1 \bullet (f[i] \geq 0 \Rightarrow (f[i-1] < 0 \Leftrightarrow f[i+1] \geq 0)) \wedge$
   $(f[i] < 0 \Rightarrow (f[i-1] \geq 0 \wedge f[i+1] \geq 0))$

Geben sie für die folgenden Aussagen die Nummer(n) von **gleichwertigen** Formeln an, Mehrfachnennungen sind möglich, gibt es keine Formel tragen Sie 0 an die vorgesehene Stelle ein. Ein Hinweis zur Semantik der Aussagen: positiv ist hier das Gegenteil von negativ.

1. Es stehen nie 3 negative Werte nebeneinander.

   .....

2. Es stehen absechselnd 2 positive und ein negativer Wert im Feld, wobei das Vorzeichen des 1. Elements nicht festgelegt ist.

   .....

3. Es stehen absechselnd 2 positive und ein negativer Wert im Feld, wobei das Vorzeichen des 1. Elements immer negativ ist.

   .....

4. Alle Werte im Feld sind positiv.

   .....

5. jeder positive Wert in f, außer den Randwerten, hat genau einen positiven Nachbarwert.

   .....

6. jeder positive Wert in f, außer den Randwerten, hat genau einen negativen Nachbarwert.

   .....

7. Es stehen nie 3 positive Werte nebeneinander.

   .....

8. jeder positive Wert in f, außer den Randwerten, hat mindestens einen negativen Nachbarwert.

   .....

Aufgabe 3:

Die logischen Operatoren $\wedge$, $\vee$ und $\Leftrightarrow$ können durch gleichwertige bedingte Ausdrücke formuliert werden:

$a \wedge b \equiv$ if a then b else false
$a \vee b \equiv$ if a then true else b
$a \Leftrightarrow b \equiv$ if a then b else $\neg b$

Verwenden Sie diese Regeln, um in der folgenden Funktion die bedingten Ausdrücke durch logische Operatoren zu ersetzen:

```
f(x : N0) : B
    if x = 0
    then false
    else
        if x mod 2 = 1
        then ¬f(x div 2)
        else f(x div 2)
```

Die transformierte Funktion:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Wann ist in einer Programmiersprache diese Transformation erlaubt?

. . . . . .

. . . . . .

Aufgabe 4:

Gegeben seinen folgende Variablen

var $x, y, z : \mathsf{Z}$

Berechnen Sie zu den folgenden Anweisungen und Nachbedingungen die zugehörigen (vereinfachten) Vorbedingungen V:

1. $\{\ V\ \}\ x, y, z := x + 1, y + x, z + y\ \{\ x = 2 \wedge y = 3 \wedge z = 5\ \}$

   .....

2. $\{\ V\ \}$ if $y > x$ then $x, y := y, x$ end if $\{\ x > y\ \}$

   .....

3. $\{\ V\ \}$ if $y > x$ then $x, y := y, x$ end if $\{\ x \geq y\ \}$

   .....

4. $\{\ V\ \}\ x := x + 1; y := y + x; z := z + y\ \{\ x = 2 \wedge y = 3 \wedge z = 5\ \}$

   .....

5. $\{\ V\ \}\ z := z + y; y := y + x; x := x + 1\ \{\ x = 2 \wedge y = 3 \wedge z = 5\ \}$

   .....

Aufgabe 5:

Gegeben sei die folgende Funktion:

```
fib(n : N0) : N0
    begin
        var i, x0, x1 : N0;
        i, x0, x1 := 0, 0, 1;
        while i < n do
            i, x0, x1 := i + 1, x1, x0 + x1
        end while ;
        x0
    end
```

Transformieren Sie diese Funktion in eine gleichwertige rekursive Form, die ohne Zuweisungen arbeitet. Hinweis: Man braucht hierzu eine Hilfsfunktion.

Aufgabe 6:

Gegeben seien die beiden Funktionen $primFaktorSumme$ und pfs.

$pfs(n : \mathsf{N}_0, t : \mathsf{N}_0) : \mathsf{N}_0$
  if $n < t$
  then 0
  else
  if n mod $t = 0$
  then $t + pfs(n \text{ div } t, t)$
  else $0 + pfs(n, t + 1)$

$primFaktorSumme(n : \mathsf{N}_0) : \mathsf{N}_0$
  $pfs(n, 2)$

Welche Resultate liefern die Aufrufe von

1. $primFaktorSumme(2)$

   .....

2. $primFaktorSumme(8)$

   .....

3. $primFaktorSumme(10)$

   .....

Transformieren Sie die Funktion pfs in eine gleichwertige Funktion, die mit einer Schleife arbeitet. Benutzen Sie hierzu die Techniken aus der Vorlesung.

Die **vollständig** transformierte Funktion: