Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

4. Februar 1997

Aufgaben zur Klausur **Grundlagen der Programmierung**, **Software Engineering** und **Logische Programmierung** im WS 96/97 (WI03)

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:

Analysieren Sie die folgenden Aussagen. Dabei ist die Grundmenge, über die Aussagen gemacht wird, die Menge aller Software, hier mit $Software$ bezeichnet, diese ist nicht leer.

Es werden folgende einstellige Elementaraussagen verwendet:

$vonWW(sw)$
: für ein Stück Software, das von der Firma *WinzigWeich* gebaut wurde

$preiswert(sw)$
: für ein Stück preiswerter Software, nicht preiswerte Software ist teuer

$fehleranfaellig(sw)$
: für fehleranfällige Software, nicht fehleranfällige Software ist zuverlässig

Die Aussagen über Software als prädikatenlogische Formeln

1. $\exists\, sw \in Software \bullet (vonWW(sw) \wedge fehleranfaellig(sw)) \Rightarrow preiswert(sw)$
2. $\exists\, sw \in Software \bullet vonWW(sw) \wedge (preiswert(sw) \wedge fehleranfaellig(sw))$
3. $\exists\, sw \in Software \bullet (\neg vonWW(sw) \wedge preiswert(sw)) \Rightarrow \neg fehleranfaellig(sw)$
4. $\exists\, sw \in Software \bullet preiswert(sw) \Rightarrow (vonWW(sw) \wedge fehleranfaellig(sw))$
5. $\exists\, sw \in Software \bullet preiswert(sw) \Rightarrow \neg(vonWW(sw) \wedge \neg preiswert(sw))$
6. $\forall\, sw \in Software \bullet vonWW(sw) \Rightarrow preiswert(sw) \wedge fehleranfaellig(sw)$
7. $\forall\, sw \in Software \bullet (\neg fehleranfaellig(sw) \vee \neg preiswert(sw)) \Rightarrow \neg vonWW(sw)$
8. $\forall\, sw \in Software \bullet \neg vonWW(sw) \vee (preiswert(sw) \wedge \neg fehleranfaellig(sw))$
9. $\forall sw \in Software \bullet fehleranfaellig(sw) \Rightarrow (vonWW(sw) \wedge \neg fehleranfaellig(sw))$
10. $\forall\, sw \in Software \bullet (\neg vonWW(sw) \vee \neg preiswert(sw)) \vee fehleranfaellig(sw)$
11. $\forall\, sw \in Software \bullet preiswert(sw) \Rightarrow (vonWW(sw) \wedge fehleranfaellig(sw))$
12. $\forall\, sw \in Software \bullet vonWW(sw) \Rightarrow (\neg preiswert(sw) \wedge fehleranfaellig(sw))$

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.

1. Es gibt WinzigWeich–Software, die ist preiswert und zuverlässig.

   .....

2. Software von WinzigWeich ist immer preiswert und zuverlässig.

   .....

3. Software von WinzigWeich ist weder preiswert noch zuverlässig.

   .....

4. Jede preiswerte Software ist von WinzigWeich und fehleranfällig.

   .....

5. Es gibt teure Software oder fehleranfällige WinzigWeich–Software.

   .....

6. Software von WinzigWeich ist immer preiswert und fehleranfällig.

   .....

7. Jede preiswerte Winzigweich–Software ist fehleranfällig.

   .....

8. Es gibt preiswerte, aber fehleranfällige WinzigWeich-Software.

   .....

9. Falsch.

   .....

10. Wahr.

    .....

Aufgabe 2:

Berechnen Sie für die folgenden Programmstücke S und die gegebenen Nachbedingungen Q mit Hilfe der Beweisregeln für Zuweisung, Anweisungsfolge und Verzweigung die zugehörige Vorbedingung P und vereinfachen Sie diese gegebenenfalls.

Dabei werden folgende Variablen verwendet:

$$\begin{array}{l}\mathsf{var}\ x, y, z : \mathsf{R};\\ \qquad b : \mathsf{B};\end{array}$$

$$\{\ P\ \}\ S\ \{\ Q\ \}$$

1. $\{\ P\ \}\ x, y, z := y, z, x\ \{\ x = w_1 \wedge y = w_2 \wedge z = w_3\ \}$

   $P(\text{vereinfacht}) :$

   .....

2. $\{\ P\ \}\ x := y;\ y := x\ \{\ x = w_1 \wedge y = w_2\ \}$

   $P(\text{vereinfacht}) :$

   .....

3. $\{\ P\ \}\ x := x - z; y := y + z\ \{\ x + y = c\ \}$

   $P(\text{vereinfacht}) :$

   .....

4. $\{\ P\ \}\ b := x > y\ \{\ b \oplus (x \leq y)\ \}$

   $P(\text{vereinfacht}) :$

   .....

5. $\{\ P\ \}\ b := x > y\ \{\ b \Leftrightarrow (x \geq y)\ \}$

   $P(\text{vereinfacht}) :$

   .....

Aufgabe 3:

Gegeben seien die folgenden Variablen

$$\begin{array}{l}\mathsf{var}\ f : \mathsf{array}\ [0..n-1]\ of\ \mathsf{R};\\ \qquad i : \mathsf{N}_0;\\ \qquad b : \mathsf{B}\end{array}$$

Entwickeln Sie für die folgende Programmspezifikation eine Programmstück, das mit einer Schleife arbeitet. Wenden Sie dabei Verfahren aus der Vorlesung an.

$\{\ \mathsf{true}\ \}$

$PraedikatBerechnen$

$\{\ b \Leftrightarrow \forall\, 1 \leq i < n \bullet f[i-1] < 0 \Leftrightarrow f[i] \geq 0\ \}$

das Programmstück $PraedikatBerechnen$:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Aufgabe 4:

Entwickeln Sie ein Prolog Prädikat $min(+X, +Y, -Min)$, das das Minimum der Zahlen X und Y berechnet. Das Prädikat soll deterministisch arbeiten und keinen *Cut* enthalten.

.....

.....

.....

.....

.....

.....

.....

Entwickeln Sie ein Prolog Prädikat $mini(+L, -Min)$, das das Minimum der Elemente einer Liste berechnet. *mini* soll ebenfalls determinsitisch arbeiten. Eine leere Liste hat kein Minimum.

.....

.....

.....

.....

.....

.....

.....

Aufgabe 5:

Gegeben sei das folgende Prolog-Prädikat $kante(Start, Ziel, Laenge)$ zur Beschreibung von gerichteten Graphen mit Start- und Zielknoten und der Länge einer Kante. Ein Prädikat weg das die Existenz von Wegen in einem Graphen überprüft hat folgendes Aussehen:

$weg(X, Y) :-$
    $kante(X, Y, \_).$

$weg(X, Y) :-$
    $kante(X, Z, \_),$
    $weg(Z, Y).$

Erweitern Sie dieses Prädikat zu einem 3-stelligen Prädikat $weg(+X, +Y, -Len)$, das die Länge eines Weges berechnet.

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Erweitern Sie das 3-stellige Prädikat zu einem 4-stelligen $weg(+X, +Y, +MaxLen, -Len)$, das testet ob es einen Weg gibt, der höchstens die Länge $MaxLen$ besitzt.

Gegeben sei der folgende Graph:

$kante(a, b, 2).$
$kante(c, b, 0).$
$kante(c, a, 3).$
$kante(b, c, 1).$

Liefert das 3-stellige weg-Prädikat für eine Anfrage $? -\ weg(a, a, L).$ für den Graphen eine Antwort?

ja ☐ nein ☐

Begründung:

. . . . . .

. . . . . .

Liefert das 4-stellige weg-Prädikat für eine Anfrage $? -\ weg(a, a, 100, []).$ für den Graphen eine Antwort?

ja ☐ nein ☐

Begründung:

. . . . . .

. . . . . .

Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

4. Februar 1997

Aufgaben zur Klausur **Grundlagen der Programmierung** und **Software Engineering** im WS 96/97 (II13)

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 8 Seiten

---

Aufgabe 1:

Analysieren Sie die folgenden Aussagen. Dabei ist die Grundmenge, über die Aussagen gemacht wird, die Menge aller Software, hier mit $Software$ bezeichnet, diese ist nicht leer.

Es werden folgende einstellige Elementaraussagen verwendet:

$vonWW(sw)$
: für ein Stück Software, das von der Firma *WinzigWeich* gebaut wurde

$preiswert(sw)$
: für ein Stück preiswerter Software, nicht preiswerte Software ist teuer

$fehleranfaellig(sw)$
: für fehleranfällige Software, nicht fehleranfällige Software ist zuverlässig

Die Aussagen über Software als prädikatenlogische Formeln

1. $\exists\, sw \in Software \bullet (vonWW(sw) \wedge fehleranfaellig(sw)) \Rightarrow preiswert(sw)$
2. $\exists\, sw \in Software \bullet vonWW(sw) \wedge (preiswert(sw) \wedge fehleranfaellig(sw))$
3. $\exists\, sw \in Software \bullet (\neg vonWW(sw) \wedge preiswert(sw)) \Rightarrow \neg fehleranfaellig(sw)$
4. $\exists\, sw \in Software \bullet preiswert(sw) \Rightarrow (vonWW(sw) \wedge fehleranfaellig(sw))$
5. $\exists\, sw \in Software \bullet preiswert(sw) \Rightarrow \neg(vonWW(sw) \wedge \neg preiswert(sw))$
6. $\forall\, sw \in Software \bullet vonWW(sw) \Rightarrow preiswert(sw) \wedge fehleranfaellig(sw)$
7. $\forall\, sw \in Software \bullet (\neg fehleranfaellig(sw) \vee \neg preiswert(sw)) \Rightarrow \neg vonWW(sw)$
8. $\forall\, sw \in Software \bullet \neg vonWW(sw) \vee (preiswert(sw) \wedge \neg fehleranfaellig(sw))$
9. $\forall sw \in Software \bullet fehleranfaellig(sw) \Rightarrow (vonWW(sw) \wedge \neg fehleranfaellig(sw))$
10. $\forall\, sw \in Software \bullet (\neg vonWW(sw) \vee \neg preiswert(sw)) \vee fehleranfaellig(sw)$
11. $\forall\, sw \in Software \bullet preiswert(sw) \Rightarrow (vonWW(sw) \wedge fehleranfaellig(sw))$
12. $\forall\, sw \in Software \bullet vonWW(sw) \Rightarrow (\neg preiswert(sw) \wedge fehleranfaellig(sw))$

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.

1. Es gibt WinzigWeich–Software, die ist preiswert und zuverlässig.

   .....

2. Software von WinzigWeich ist immer preiswert und zuverlässig.

   .....

3. Software von WinzigWeich ist weder preiswert noch zuverlässig.

   .....

4. Jede preiswerte Software ist von WinzigWeich und fehleranfällig.

   .....

5. Es gibt teure Software oder fehleranfällige WinzigWeich–Software.

   .....

6. Software von WinzigWeich ist immer preiswert und fehleranfällig.

   .....

7. Jede preiswerte Winzigweich–Software ist fehleranfällig.

   .....

8. Es gibt preiswerte, aber fehleranfällige WinzigWeich-Software.

   .....

9. Falsch.

   .....

10. Wahr.

    .....

Aufgabe 2:

Berechnen Sie für die folgenden Programmstücke S und die gegebenen Nachbedingungen Q mit Hilfe der Beweisregeln für Zuweisung, Anweisungsfolge und Verzweigung die zugehörige Vorbedingung P und vereinfachen Sie diese gegebenenfalls.

Dabei werden folgende Variablen verwendet:

$$\begin{array}{l}\mathsf{var}\ x, y, z : \mathsf{R};\\ \qquad b : \mathsf{B};\end{array}$$

$$\{\ P\ \}\ S\ \{\ Q\ \}$$

1. $\{\ P\ \}\ x, y, z := y, z, x\ \{\ x = w_1 \wedge y = w_2 \wedge z = w_3\ \}$

   $P(\text{vereinfacht}) :$

   .....

2. $\{\ P\ \}\ x := y;\ y := x\ \{\ x = w_1 \wedge y = w_2\ \}$

   $P(\text{vereinfacht}) :$

   .....

3. $\{\ P\ \}\ x := x - z; y := y + z\ \{\ x + y = c\ \}$

   $P(\text{vereinfacht}) :$

   .....

4. $\{\ P\ \}\ b := x > y\ \{\ b \oplus (x \leq y)\ \}$

   $P(\text{vereinfacht}) :$

   .....

5. $\{\ P\ \}\ b := x > y\ \{\ b \Leftrightarrow (x \geq y)\ \}$

   $P(\text{vereinfacht}) :$

   .....

Aufgabe 3:

Gegeben seien die folgenden Variablen

$$\begin{array}{l}\mathsf{var}\ f : \mathsf{array}\ [0..n-1]\ of\ \mathsf{R};\\ \qquad i : \mathsf{N}_0;\\ \qquad b : \mathsf{B}\end{array}$$

Entwickeln Sie für die folgende Programmspezifikation eine Programmstück, das mit einer Schleife arbeitet. Wenden Sie dabei Verfahren aus der Vorlesung an.

$\{\ \mathsf{true}\ \}$

$PraedikatBerechnen$

$\{\ b \Leftrightarrow \forall\, 1 \leq i < n \bullet f[i-1] < 0 \Leftrightarrow f[i] \geq 0\ \}$

das Programmstück $PraedikatBerechnen$:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Aufgabe 4:

Gegeben sei die folgende Funktion f

$f(i : \mathsf{N}_0) : \mathsf{R}$
  if $i = 0$
  then $g(i)$
  else $h_1(i) + f(h_2(i))$

Diese benutzt Funktionen g, h_1 und h_2. Die Funktionsköpfe diese Funktionen haben folgende Gestalt:

$g(n : \mathsf{N}_0) : \mathsf{R}$
$h_1(n : \mathsf{N}_0) : \mathsf{R}$
$h_2(n : \mathsf{N}_0) : \mathsf{N}_0$

Transformieren Sie die Funktion f gemäß der Transformationsschemata aus der Vorlesung in eine gleichwertige Funktion, die mit einer Schleife arbeitet.

$f(i : \mathsf{N}_0) : \mathsf{R}$

Aufgabe 5:

Beweisen Sie durch Transformation, daß die folgende Formel ein Satz der Aussagenlogik ist. Begründen Sie die einzelnen Beweisschritte.

$$(x \Rightarrow y) \wedge \neg(x \oplus z) \Rightarrow (\neg y \Rightarrow \neg z)$$

Nutzen Sie diese Seite für die Kladde, die nächste Seite für die fertige Lösung.

$$(x \Rightarrow y) \wedge \neg(x \oplus z) \Rightarrow (\neg y \Rightarrow \neg z)$$

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....

$\Leftrightarrow$ *Begründung :* .....

.....