Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

3. Februar 1995

---

Aufgaben zur Klausur **Grundlagen der Programmierung**, **Software Engineering** und **Logische Programmierung** im WS 94/95 (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.

Viel Erfolg !

Diese Klausur besteht einschließlich dieses Deckblattes aus 6 Seiten

---

**Aufgabe 1:**

Entwerfen Sie einen Algorithmus (eine Funktion) mit Namen $ps$, der für eine reelle Zahl $x$ und eine natürliche Zahl $n$ die Summe der Potenzen $x^i$ berechnet für $0 \leq i < n$, also $1 + x + x^2 + \ldots + x^{n-1}$.

Nutzen Sie hierbei die Eigenschaft aus, daß
$1 + x + x^2 + \ldots + x^{n-1} \;=\; 1 + x * (1 + x + \ldots + x^{n-2})$
ist.

Die Funktion $ps$ soll ohne Zuweisungen arbeiten, also als Rumpf nur einen Ausdruck enthalten.

$ps(x : \mathsf{R}, n : \mathsf{N}_0) : \mathsf{R}$

.....

.....

.....

.....

.....

Ist dieser Algorithmus zur Parallelverarbeitung geeignet?

ja ☐ nein ☐

Begründung:

.....

Ist dieser Algorithmus einfach in eine Schleife zu transformieren, da er mit einer Endrekursion arbeitet?

ja ☐ nein ☐

Begründung:

.....

**Aufgabe 2:**

Gegeben sei ein Feld $f$

var $f$ : array $[0..n-1]$ of $\mathsf{N}_0$

Welche der folgenden prädikatenlogischen Formeln sagen aus, daß $f$ absteigend sortiert ist und Duplikate nicht zugelassen sind.

1. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet i < j \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
2. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet i \leq j \Rightarrow f[i] \leq f[j]$ ja ☐ nein ☐ weiß nicht ☐
3. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet j < i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
4. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet j \leq i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
5. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet j > i \Rightarrow f[j] \leq f[i]$ ja ☐ nein ☐ weiß nicht ☐
6. $\forall\, 0 \leq i < n \bullet \forall\, i < j < n \bullet j > i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
7. $\forall\, 0 \leq j < n \bullet \forall\, 0 \leq i < n \bullet j < i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
8. $\forall\, 0 \leq i < n \bullet \forall\, i < j < n \bullet j \neq i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
9. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet f[i] \leq f[j]$ ja ☐ nein ☐ weiß nicht ☐
10. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet f[j] \leq f[i]$ ja ☐ nein ☐ weiß nicht ☐
11. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet f[j] < f[i]$ ja ☐ nein ☐ weiß nicht ☐
12. $\forall\, 0 \leq i < n \bullet \forall\, i < j < n \bullet f[j] < f[i]$ ja ☐ nein ☐ weiß nicht ☐

---

**Aufgabe 3:**

Zeigen Sie durch Transformation, daß die beiden Aussagen $x \vee y \Rightarrow x \wedge y$ und $x \Leftrightarrow y$ äquivalent sind, d.h. daß $x \vee y \Rightarrow x \wedge y \Leftrightarrow (x \Leftrightarrow y)$ ein Satz ist. Geben Sie in den mit *Begründung:* markierten Zeilen jeweils die verwendeten Gesetze oder deren Namen an.

$x \vee y \Rightarrow x \wedge y$

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

.....

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

.....

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

.....

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

.....

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

.....

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

.....

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

$x \Leftrightarrow y$

**Aufgabe 4:**

Schreiben Sie ein 2-stelliges Prädikat $rmduplicates(L, R)$, das aus der Liste $L$ eine neue Liste $R$ aufbaut, in der alle Elemente von $L$ enthalten sind, aber keine Duplikate vorkommen.

Benutzen Sie hierzu ein Prädikat $insert(E, S, S1)$, das das Element $E$ zu der Liste $S$ hinzufügt und in $S1$ zurückgibt, falls es noch nicht enthalten ist. Dieses Prädikat soll deterministisch arbeiten. Das *member*–Prädikat darf verwendet werden. Verwenden Sie, falls notwendig, keinen Cut sondern das *not* Prädikat.

$rmduplicates(+L, -R)$ :

.....

.....

.....

.....

.....

$insert(+E, +S, -S1)$ :

.....

.....

.....

.....

.....

Hat die Verwendung des *not* Prädikats gegenüber dem Cut Vorteile?

ja ☐ nein ☐

Begründung:

.....

.....

**Aufgabe 5:**

Schreiben Sie in PROLOG ein 3-stelliges Prädikat $occurs(E, L, N)$. Das 1. Argument $E$ soll einen beliebigen Wert aufnehmen, das 2. Argument $L$ eine Liste und das 3. Argument $N$ eine Zahl. $occurs$ soll in $N$ berechnen, wie häufig $E$ als Element in der Liste $L$ vorkommt. Verwenden sie keinen CUT.

.....

.....

.....

.....

.....

.....

Was antwortet das System auf die folgenden Anfragen?

1. $occurs(eins, [], 0)$

   .....

   .....

2. $occurs(eins, [eins, zwei, eins], 1)$

   .....

   .....

3. $occurs(\_, [eins, zwei, drei], N)$

   .....

   .....

4. $occurs(\_, \_, N)$

   .....

   .....

Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

3. Februar 1995

---

Aufgaben zur Klausur **Grundlagen der Programmierung** und **Software Engineering** im WS 94/95 (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.

Viel Erfolg !

Diese Klausur besteht einschließlich dieses Deckblattes aus 5 Seiten

---

**Aufgabe 1:**

Entwerfen Sie einen Algorithmus (eine Funktion) mit Namen $ps$, der für eine reelle Zahl $x$ und eine natürliche Zahl $n$ die Summe der Potenzen $x^i$ berechnet für $0 \leq i < n$, also $1 + x + x^2 + \ldots + x^{n-1}$.

Nutzen Sie hierbei die Eigenschaft aus, daß
$1 + x + x^2 + \ldots + x^{n-1} \;=\; 1 + x * (1 + x + \ldots + x^{n-2})$
ist.

Die Funktion $ps$ soll ohne Zuweisungen arbeiten, also als Rumpf nur einen Ausdruck enthalten.

$ps(x : \mathsf{R}, n : \mathsf{N}_0) : \mathsf{R}$

.....

.....

.....

.....

.....

Ist dieser Algorithmus zur Parallelverarbeitung geeignet?

ja ☐ nein ☐

Begründung:

.....

Ist dieser Algorithmus einfach in eine Schleife zu transformieren, da er mit einer Endrekursion arbeitet?

ja ☐ nein ☐

Begründung:

.....

**Aufgabe 2:**

Gegeben sei ein Feld $f$

var $f$ : array $[0..n-1]$ of $\mathsf{N}_0$

Welche der folgenden prädikatenlogischen Formeln sagen aus, daß $f$ absteigend sortiert ist und Duplikate nicht zugelassen sind.

1. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet i < j \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
2. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet i \leq j \Rightarrow f[i] \leq f[j]$ ja ☐ nein ☐ weiß nicht ☐
3. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet j < i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
4. $\forall\, 0 \leq i < n \bullet \forall\, 0 \leq j < n \bullet j \leq i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
5. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet j > i \Rightarrow f[j] \leq f[i]$ ja ☐ nein ☐ weiß nicht ☐
6. $\forall\, 0 \leq i < n \bullet \forall\, i < j < n \bullet j > i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
7. $\forall\, 0 \leq j < n \bullet \forall\, 0 \leq i < n \bullet j < i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
8. $\forall\, 0 \leq i < n \bullet \forall\, i < j < n \bullet j \neq i \Rightarrow f[i] < f[j]$ ja ☐ nein ☐ weiß nicht ☐
9. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet f[i] \leq f[j]$ ja ☐ nein ☐ weiß nicht ☐
10. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet f[j] \leq f[i]$ ja ☐ nein ☐ weiß nicht ☐
11. $\forall\, 0 \leq i < n \bullet \forall\, i \leq j < n \bullet f[j] < f[i]$ ja ☐ nein ☐ weiß nicht ☐
12. $\forall\, 0 \leq i < n \bullet \forall\, i < j < n \bullet f[j] < f[i]$ ja ☐ nein ☐ weiß nicht ☐

---

**Aufgabe 3:**

Zeigen Sie durch Transformation, daß die beiden Aussagen $x \vee y \Rightarrow x \wedge y$ und $x \Leftrightarrow y$ äquivalent sind, d.h. daß $x \vee y \Rightarrow x \wedge y \Leftrightarrow (x \Leftrightarrow y)$ ein Satz ist. Geben Sie in den mit *Begründung:* markierten Zeilen jeweils die verwendeten Gesetze oder deren Namen an.

$x \vee y \Rightarrow x \wedge y$

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

.....

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

.....

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

.....

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

.....

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

.....

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

.....

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

$x \Leftrightarrow y$

**Aufgabe 4:**

Gegeben seien die ganzzahligen Variablen $a, b, c$ und $d$ und eine Boolesche Variable $r$.

Transformieren Sie das folgende Programmstück in ein gleichwertiges, das nur noch aus einer Zuweisung besteht und höchstens drei arithmetische Vergleiche enthält.

```
if ((a > 0) ∧ (b ≤ 0))
      ∨ ((a ≤ 0) ∧ (b > 0))
then
      if c ≥ d
      then r := false
      else r := true
      end if
else
      if c < d
      then r := false
      else r := true
      end if
end if
```

. . . . . .

. . . . . .

---

FACHHOCHSCHULE WEDEL
Prof. Dr. Uwe Schmidt

3. Februar 1995

---

Aufgaben zur Klausur **Logische Programmierung** im WS 94/95 (IA41)

Zeit: 60 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.

Viel Erfolg !

Diese Klausur besteht einschließlich dieses Deckblattes aus 4 Seiten

---

**Aufgabe 1:**

Gegeben seien die folgenden Prolog–Klauseln

$kante(a, b).$
$kante(b, c).$
$kante(a, d).$
$kante(b, d).$
$kante(c, d).$
$kante(d, e).$

$weg(X, Y, [X, Y]) :-$
$\quad kante(X, Y).$
$weg(X, Y, [X|W]) :-$
$\quad kante(X, Z), weg(Z, Y, W).$

Welche Antwort(en) erhält man mit der Anfrage

$weg(a, e, W)$

Wenn es mehrere Antworten gibt, geben Sie sie bitte in der Reihenfolge an, in der sie auch erzeugt werden.

1) .....

2) .....

3) .....

4) .....

5) .....

6) .....

Welche Antwort(en) erhält man mit derselben Anfrage, wenn die Datenbasis am Ende um den Fakt

$kante(b, a).$

erweitert wird.

1) .....

2) .....

3) .....

4) .....

5) .....

6) .....

---

**Aufgabe 2:**

Schreiben Sie ein 2-stelliges Prädikat $rmduplicates(L, R)$, das aus der Liste $L$ eine neue Liste $R$ aufbaut, in der alle Elemente von $L$ enthalten sind, aber keine Duplikate vorkommen.

Benutzen Sie hierzu ein Prädikat $insert(E, S, S1)$, das das Element $E$ zu der Liste $S$ hinzufügt und in $S1$ zurückgibt, falls es noch nicht enthalten ist. Dieses Prädikat soll deterministisch arbeiten. Das *member*–Prädikat darf verwendet werden. Verwenden Sie, falls notwendig, keinen Cut sondern das *not* Prädikat.

$rmduplicates(+L, -R)$ :

.....

.....

.....

.....

.....

$insert(+E, +S, -S1)$ :

.....

.....

.....

.....

.....

Hat die Verwendung des *not* Prädikats gegenüber dem Cut Vorteile?

ja ☐ nein ☐

Begründung:

.....

.....