Übung 1: Boolesche Algebra (vorbereitender Überblick)

Im folgenden sind x,y und z Variablen mit Werten in B={wahr, falsch}.

0. Vervollständigen Sie die folgende Wahrheitstabelle:

x	y	x und y	x oder y	nicht x	$x \Rightarrow y$	$x \Leftarrow y$	$x \Leftrightarrow y$	x = y
falsch	falsch							
falsch	wahr							
wahr	falsch							
wahr	wahr							

1. Die Funktionen fn1, fn2 und fn3 sind wie folgt definiert:

x	y	x fn1 y	x fn2 y	x fn3 y
falsch	falsch	wahr	falsch	falsch
falsch	wahr	falsch	wahr	wahr
wahr	falsch	falsch	falsch	wahr
wahr	wahr	falsch	falsch	falsch

Schreiben Sie für jede dieser Funktionen einen für alle x und y gleichwertigen Ausdruck. Verwenden Sie dabei nur die Booleschen Funktionen und, oder und nicht.

1.0. (x fn1 y) = ?
1.1. (x fn2 y) = ?
1.2. (x fn3 y) = ?

1.3. Wieviele verschiedene Funktionen von $B \times B \rightarrow B$ gibt es? Von $B \times B \times B \rightarrow B$? Von $B^n \rightarrow B$?

1.4. Können sie alle unter Verwedung nur der Funktionen und, oder und nicht geschrieben werden? Wenn ja, wie? Wenn nicht, Gegenbeispiel angeben.

2. In Ausdrücken werden die verschiedenen Funktionen üblicherweise in der folgenden Reihenfolge angewendet, wenn eine andere nicht durch Klammern explizit angegeben ist:

↑	(potenzieren)
+, -	(Vorzeichen)
*, /	(Multiplikation, Division)
+, -	(Addition, Subtraktion)
<, >, =, ≤, ≥, ≠	(relationale Funktionen)
nicht	
und	
oder	
⇒, ⇐, ⇔	(logische Implikationen)

Beweisen Sie die folgenden Aussagen (Gleichungen) für alle x,y,z∈ B:

2.0. (x und y) = (y und x)
2.1. (x oder y) = (y oder x)

2.2. (x und (y und z)) = ((x und y) und z)
2.3. (x oder (y oder z)) = ((x oder y) oder z)

2.4. (x und (y oder z)) = ((x und y) oder (x und z))
2.5. (x oder (y und z)) = ((x oder y) und (x oder z))

2.6. (x und nicht x) = falsch
2.7. (x und falsch) = falsch
2.8. (x und x) = x
2.9. (x und wahr) = x

2.10. (x oder falsch) = x
2.11. (x oder x) = x
2.12. (x oder wahr) = wahr
2.13. (x oder nicht x) = wahr

2.14. (x oder (x und y)) = x
2.15. (x oder (nicht x und y)) = (x oder y)

2.16. (nicht (nicht x)) = x

2.17. (nicht (x und y)) = ((nicht x) oder (nicht y))
2.18. (nicht (x oder y)) = ((nicht x) und (nicht y))

2.19. (x $\Rightarrow$ y) = (nicht (x und nicht y))
2.20. (x $\Rightarrow$ y) = ((nicht x) oder y)
2.21. (x $\Rightarrow$ y) = ((nicht y) $\Rightarrow$ (nicht x))
2.22. (z und (x $\Rightarrow$ y)) = (z und ((z und x) $\Rightarrow$ y))

2.23. (x = y) = ((x und y) oder (nicht x und nicht y))

(Ende)

Übung 2: Vereinfachung Boolescher Ausdrücke

0. Seien B, C und D Boolesche Funktionen (d.h., Funktionen mit Werten in B). Die Funktion wird wie folgt definiert:

F = C, falls B = wahr
= D, falls B = falsch

Schreiben Sie für F einen gleichwertigen Ausdruck. Verwenden Sie dabei außer B, C und D nur die Booleschen Funktionen und, oder und nicht. Zeigen Sie, daß Ihr Ausdruck der oben stehenden Definition von F genügt.

1. Vereinfachen bzw. expandieren Sie die folgenden Ausdrücke:

1.0. x und (y $\Rightarrow$ z)
1.1. x oder (y $\Rightarrow$ z)
1.2. (x und y) $\Rightarrow$ z
1.3. (x oder y) $\Rightarrow$ z

1.4. -x>0 und x<0 oder x>0 und nicht x<0
1.5. -x≥0 und x<0 oder x≥0 und nicht x<0
1.6. -x<0 und x<0 oder x<0 und nicht x<0
1.7. -x≤0 und x<0 oder x≤0 und nicht x<0

1.8. (w oder x) und (y oder z)
1.9. [ia≤na oder ib≤nb] und [ib>nb oder ia≤na und A(ia)≤B(ib)]
1.10. [ia≤na oder ib≤nb] und [ib>nb oder ia≤na und ib≤nb und A(ia)≤B(ib)]
1.11. nicht (ib>nb oder ia≤na und A(ia)≤B(ib))

2. Zeigen Sie, daß

{nicht (ia≤na und [ib>nb oder A(ia)≤B(ib)])}
= {ib≤nb und [ia>na oder A(ia)>B(ib)]}

falls ia≤na oder ib≤nb.

(Ende)

Übung 3: Werte von Variablen und Ausdrücken in einer Datenumgebung

0. Sei die Datenumgebung d=[(x, R, 9.3), (y, R, 2), (z, Z, 3),(z, Z, 4)]. Bestimmen Sie die Werte der folgenden Variablen bzw. Ausdrücke in d:

0.0. z
0.1. y
0.2. x
0.3. w
0.4. 6.6
0.5. x+2*y
0.6. y↑z - x
0.7. y↑x - r

1. Sei die Datenumgebung

d = [(k, Z, 2), (y(0), R, 2), (y(1), R, 3.2), (j, Z, 1), (y(2), R, 5.1), (j, Z, 3), (y(3), R, 7.5), (y(2), Z, 9)]

Bestimmen Sie die Werte der folgenden Variablen bzw. Ausdrücke in d:

1.0. y(k)
1.1. y(j+k)
1.2. y(k-j) + (j-k)
1.3. y(j-k)
1.4. y(k-j+x)
1.5. y(3*j-k) + y(2*j)
1.6. y(k/2)
1.7. y((j+k)/2)
1.8. y((j+k+1)/2)
1.9. y(k) < y(j+k)
1.10. y(k) ≥ y(j-k)
1.11. j ≤ k

2. Ein Variablenname (z.B. x) kann als Funktion auf D aufgefaßt werden. Beschreiben Sie den Definitionsbereich dieser Funktion. Was ist ihr Wertebereich?

3. Ein Ausdruck (z.B. x/(y-3)>z und nm = "AP136) ist im allgemeinen eine Zusammensetzung von Funktionen und kann selbst als Funktion auf D aufgefaßt werden. Beschreiben Sie den Definitionsbereich dieser Funktion. Was ist ihr Wertebereich?

(Ende)

Übung 4: Zuweisungen, if-Anweisungen und Folgen als Funktionen auf D

0. Sei die Datenumgebung

d0 = [(x, R, 9.3), (y, R, 2), (z, Z, 3), (z, Z, 4), (w, Z, 6)].

Bestimmen Sie die Datenumgebungen d1 = Z(d0) für jede der folgenden Zuweisungen Z:

0.0. x := 1.4*y + z
0.1. z := z + y
0.2. z := x + y
0.3. (z, w) := (y + z + w, 2*y + 3*z + 4*w)
0.4. (z, z) := (y + z + w, 2*y + 3*z + 4*w)
0.5. (z, z) := (y + z + w, y*w - 1)
0.6. (y, w) := (w, y)

1. Sei die Datenumgebung

d = [(k, Z, 2), (y(0), R, 2), (y(1), R, 3.2), (j, Z, 1), (y(2), R, 5.1),
(j, Z, 3), (y(3), R, 7.5), (y(2), Z, 9)]

Bestimmen Sie die Datenumgebung d1=Z(d0) für jede der folgenden Zuweisungen Z:

1.0. y(j) := y(k) + 3
1.1. (j, y(j)) := (j+1, y(k) + 3)

2. Sei die Datenumgebung

d0 = [(x, R, 9.3), (y, R, 2), (z, Z, 3), (w, Z, 6)].

Bestimmen Sie die Datenumgebung d1=S(d0) für jede der folgenden if-Anweisungen S:

2.0. if x<0 then y:=-x else y:=x endif
2.1. if x>0 then y:=-x else y:=x endif
2.2. if r>0 then y:=-x else y:=x endif
2.3. if x>0 then r:=-x else y:=x endif
2.4. if x<0 then r:=-x else y:=x endif

3. Bestimmen Sie die Datenumgebung d1=S(d0) für jede der unten stehenden Folgen S von Anweisungen. (Die Datenumgebung d0 ist wie in Aufgabe 2 oben angegeben).

3.0. z := y+z+w
w:= 2*y + 3*z + 4*w

3.1. x := y*z
y := w

3.2. y := w
x := y*z

3.3.
```
x := x+w*z
y := w
```

3.4.
```
y := w
x := x+w*z
```

3.5.
```
x := y*z
y := w
z := 2*x - w
```

4. Beschreiben Sie präzis den Definitionsbereich einer einfachen Zuweisung.

5. Beschreiben Sie präzis den Definitionsbereich einer mehrfachen Zuweisung.

6. Beschreiben Sie präzis den Definitionsbereich einer if-Anweisung

7. Beschreiben Sie präzis den Definitionsbereich einer Folge von zwei Anweisungen. Von mehr als zwei Anweisungen.

(Ende)

Übung 5: while-Anweisungen, Deklarationen und Löschanweisungen

0. Sei die Datenumgebung

d0 = [(i, Z, 0), (s, R, 0), (n, Z, 2), (x(1), Z, 1), (x(2), Z, 2)]

sowie die while-Anweisung W wie folgt:

```
while i<n do
        i := i + 1
        s := s + x(i)
endwhile
```

Berechnen Sie die Datenumgebung d1=W(d0) durch direkte Anwendung der formalen Definition des Ergebnisses der Anwendung einer whild-Anweisung auf eine Datenumgebung. Ermitteln Sie auch jedes Zwischenergebnis (Datenumgebung).

1. Beschreigen Sie präzis den Definitionsbereich einer while-Anweisung

2. Sei die Datenumgebung

d0 = [(x, R, 9.3), (y, R, 2)]

Bestimmen Sie die Datenumgebung d1=D(d0) für jede der folgenden Deklarationen D:

2.0. declare (z, R, x+y)
2.1. declare (z, R, z+y)

3. Beschreiben Sie präzis den Definitionsbereich einer Deklarationsanweisung. In welcher Beziehung steht er zum Definitionsbereich der entsprechenden Zuweisung?

4. Sei die Datenumgebung d0 wie in Aufgabe 2 oben angegeben. Bestimmen Sie die Datenumgebung d1=S(d0) für jede der unten stehenden Folgen S:

4.0.
```
declare (z, R, x+y)
release z
```

4.1.
```
declare (z, R, x+2*y)
release y
```

4.2.
```
declare (z, R, x)
release x
```

5. Beschreiben Sie präzis den Definitionsbereich einer release-Anweisung.

(Ende)

Übung 6: Vor- und Postbedingungen bezüglich Zuweisungen, if-Anweisungen und Folgen

0. Bestimmen Sie die jeweilige umfassende Vorbedingung:

0.0. $\{?\}\ z := z + x\ \{z \leq max\}$
0.1. $\{?\}\ x := z - y\ \{x-y > 0\}$
0.2. $\{?\}\ x := z - y\ \{y-x > 0\}$
0.3. $\{?\}\ x := 5 - z\ \{w*y - 2*w^2 < z\}$
0.4. $\{?\}$ sum := sum + z $\{sum = x + y + z\}$
0.5. $\{?\}$ y(m) := z $\{y(m) = y(n)\}$
0.6. $\{?\}$ d(j) := a(k) $\{und_{i=1}^{j-1}\ d(i) \leq d(i+1)\}$
0.7. $\{?\}$ if x < 0 then y := -x else y := x endif $\{y > 0\}$
0.8. $\{?\}$ if x < 0 then y := -x else y := x endif $\{y \geq 0\}$
0.9. $\{?\}$ if x < 0 then y := -x else y := x endif $\{y < 0\}$
0.10. $\{?\}$ if x < 0 then y := -x else y := x endif $\{y \leq 0\}$
0.11. $\{?\}$ if x < 0 then y := -x else y := x endif $\{2 \leq y \leq 4\}$
0.12. $\{?\}$ if x < 0 then y := x else y := x - 2 endif $\{-1 \leq y \leq 4\}$

0.13. $\{?\}$
i := i + 1
sum := sum + x(i)
$\{sum = \Sigma_{j=1}^{i} x(j)$ und $i \leq n$ und $i \in Z$ und $n \in Z\}$

0.14. $\{?\}$
x(gl) :=: x(gr)
gr := gr - 1
gl := gl - 1
$\{gl < gr \leq ig$ $und_{i=gl+1}^{gr}\ x(i) = x(gr)$ $und_{i=gr+1}^{ig}\ x(gr) < x(i)\}$

1. Beweisen Sie die folgenden Aussagen:

1.0. {V1 und B] S1 {P} und {nicht B} S2 {P}
$\Rightarrow$ {V1} if B then S1 else S2 endif {P}

1.1. {B} S1 {P} und {V2 und nicht B} S2 {P}
$\Rightarrow$ {V2} if B then S1 else S2 endif {P}

1.1. {V1} S1 {P} und {V2} S2 {P}
$\Rightarrow$ {V1 und V2} if B then S1 else S2 endif {P}

(Ende)

Übung 7: while-Schleife, Schleifeninvariante

0. Formulieren Sie einen möglichst starken Satz über die Korrektheit des folgenden Programmsegments:

```
declare (sum, R, 0)
declare (i, Z, 0)
while i<n do
        i := i+1
        sum := sum + x(i)
endwhile
release i
```

Der Entwickler hat als Schleifeninvariante

$$I = [sum = \Sigma_{j=1}^{i} x(k) \text{ und } 0 \leq i \leq n \text{ und } i \in Z \text{ und } n \in Z]$$

festgelegt.

1. Beweisen Sie Ihren Satz über die Korrektheit des oben angegebenen Programmsegments. Wenden Sie dabei den Schleifensatz an. Achten Sie auf alle fünf Beweisführungsschritte.

2. Leiten Sie eine umfassende Vorbedingung Ihrer Postbedingung bezüglich des oben angegebenen Programmsegments ab.

(Ende)

Übung 8: Semantik eines Pascal- und eines Basic-Programms

0. Übersetzen Sie das folgende in Pascal geschriebene Programm. Verwenden Sie in der neuen Formulierung nur die Programmanweisungen und Zusammensetzungen, die in den Vorlesungen definiert wurden (Zuweisung, Deklaration, Löschanweisung, Nullanweisung, if-Anweisung, Folge von Anweisungen, while-Schleife und Prozeduraufruf ohne Parameter).

```
program printsquares;
procedure square (par1: integer; var par2: integer);
begin
        par2 := par1 * par1;
        par1 := -10
end;

var n, f: integer;
begin
        writeln (n, ' ', f, ' Erste Zeile');
        n := 1;
        while n ≤ 10 do
        begin
                square (n, f);
                writeln (n, ' ', f);
                n := n + 1
        end
end.
```

1. Welche Auswirkungen hätten die Änderungen der zweiten Zeile oben in

```
procedure square (var par1: integer; var par2: integer);
```

und

```
procedure square (par1: integer; par2: integer);
```

auf die übersetzte Version des Programms?

2. Übersetzen Sie auf gleiche Weise das folgenen in Basic geschriebene Programm.

```
1000 REM Programmanfang: 13 Streichhölzer
1010 N=13
1020 S=1
1030 W=1
2000 IF N≤0 THEN GOTO 4000
2010 GOSUB 5000
3000 N=N-W
3010 S=3-S
3090 GOTO 2000
4000 PRINT "Das Spiel ist zu Ende. Spieler";S;"hat gewonnen."
4010 END
5000 ...
...
5080 W=...
5090 RETURN
```

(Ende)

Übung 9: Boolesche Funktionen fortgesetzt auf Be

Die Booleschen Funktionen und, oder und nicht können auf verschiedene sinnvolle Weisen auf die Menge Be = {wahr, falsch, undef} fortgesetzt werden. Dabei stellt das (wohl definierte) Element "undef" die Situation dar, wo im Falle der nicht fortgesetzten Funktionen Argument- bzw. Funktionswerte nicht definiert sind. Die am häufigsten vorkommenden Fortsetzungen dieser Funktionen werden wie folgt definiert:

x	y	x **und1** y	x **und2** y	x **und3** y
undef	undef	undef	undef	undef
undef	falsch	undef	undef	falsch
undef	wahr	undef	undef	undef
falsch	undef	undef	falsch	falsch
falsch	falsch	falsch	falsch	falsch
falsch	wahr	falsch	falsch	falsch
wahr	undef	undef	undef	undef
wahr	falsch	falsch	falsch	falsch
wahr	wahr	wahr	wahr	wahr

x	y	x **oder1** y	x **oder2** y	x **oder3** y
undef	undef	undef	undef	undef
undef	falsch	undef	undef	undef
undef	wahr	undef	undef	wahr
falsch	undef	undef	undef	undef
falsch	falsch	falsch	falsch	falsch
falsch	wahr	wahr	wahr	wahr
wahr	undef	undef	wahr	wahr
wahr	falsch	wahr	wahr	wahr
wahr	wahr	wahr	wahr	wahr

x	y	$\overset{1}{x \Rightarrow y}$	$\overset{2}{x \Rightarrow y}$	$\overset{3}{x \Rightarrow y}$
undef	undef	undef	undef	undef
undef	falsch	undef	undef	undef
undef	wahr	undef	undef	wahr
falsch	undef	undef	wahr	wahr
falsch	falsch	wahr	wahr	wahr
falsch	wahr	wahr	wahr	wahr
wahr	undef	undef	undef	undef
wahr	falsch	falsch	falsch	falsch
wahr	wahr	wahr	wahr	wahr

x	**nicht** x
undef	undef
falsch	wahr
wahr	falsch

0. Welche der folgenden Aussagen (Gleichungen) gelten für welche Fortsetzungen (Schema 1, 2 bzw. 3 oben) der Booleschen Funktionen auf **Be**? (Vgl. Übung 1, Aufgabe 2.)

0.0. (x und y) = (y und x)
0.1. (x oder y) = (y oder x)

0.2. (x und (y und z)) = ((x und y) und z)
0.3. (x oder (y oder z)) = ((x oder y) oder z)

0.4. (x und (y oder z)) = ((x und y) oder (x und z))
0.5. (x oder (y und z)) = ((x oder y) und (x oder z))

0.6. (x und nicht x) = falsch
0.7. (x und falsch) = falsch
0.8. (x und x) = x
0.9. (x und wahr) = x

0.10. (x oder falsch) = x
0.11. (x oder x) = x
0.12. (x oder wahr) = wahr
0.13. (x oder nicht x) = wahr

0.14. (x oder (x und y)) = x
0.15. (x oder (nicht x und y)) = (x oder y)
0.16. ((x oder y) und (x oder z)) = (x oder y und z)

0.17. (nicht (nicht x)) = x

0.18. (nicht (x und y)) = ((nicht x) oder (nicht y))
0.19. (nicht (x oder y)) = ((nicht x) und (nicht y))

0.20. (x $\Rightarrow$ y) = (nicht (x und nicht y))
0.21. (x $\Rightarrow$ y) = ((nicht x) oder y)
0.22. (x $\Rightarrow$ y) = ((nicht y) $\Rightarrow$ (nicht x))
0.23. (z und (x $\Rightarrow$ y)) = (z und ((z und x) $\Rightarrow$ y))

0.24. (x = y) = ((x und y) oder (nicht x und nicht y))

Literaturhinweis:

Bijlsma, A., "Semantics of Quasi-Boolean Expressions", S. 27-35 in Feijen, W.H.J., van Gasteren, A.J.M., Gries, D., and Misra, J. (eds.), *Beauty is Our Business*, *A Birthday Salute to Edsger W. Dijkstra*, Springer-Verlag, New York, 1990.

(Ende)

Übung 21: Verifikation einer Korrektheitsaussage

Beweisen Sie die folgende Korrektheitsaussage. Teilen Sie Ihren Beweis in die Schritte ein, worauf die unten stehenden Fragen hinweisen. D.h., zerlegen Sie zuerst mit Hilfe der geeigneten Beweisregeln die Korrektheitsaussage in Boolesch algebraische Ausdrücke (Lemmata), deren Wahrheit die Gültigkeit der Korrektheitsaussage sicherstellt. Danach verifizieren Sie die Lemmata.

```
{V}
i:=0; j:=0
while i<N und j≤M-N do
      if D(j+i)=K(i) then I:=i+1 else j:=j+1; i:=0 endif
invariant I
endwhile
{P}
```

wobei V, P und I wie folgt definiert sind:

V: $N \in Z$ **und** $M \in Z$ **und** $0 \le N \le M+1$

P: $N \in Z$ **und** $M \in Z$ **und** $j \in Z$
und $\mathbf{und}_{k=0}^{j-1}$ ($\mathbf{oder}_{a=0}^{N-1}$ $D(k+a) \neq K(a)$)
und ($j=M-N+1$ **oder** $0 \le j \le M-N$ $\mathbf{und}_{a=0}^{N-1}$ $D(j+a)=K(a)$)

I: $N \in Z$ **und** $M \in Z$ **und** $j \in Z$ **und** $i \in Z$ **und** $0 \le i \le N$
und $\mathbf{und}_{k=0}^{j-1}$ ($\mathbf{oder}_{a=0}^{N-1}$ $D(k+a) \neq K(a)$)
und ($j=M-N+1$ **oder** $0 \le j \le M-N$ $\mathbf{und}_{a=0}^{i-1}$ $D(j+a)=K(a)$)

Fragen:

1. Geben Sie die an jeder Stelle des Programmsegments herrschende Zwischenbedingung an. D.h., geben Sie alle für den Korrektheitsbeweis benötigten Zwischenbedingungen an.

2. Welche Boolesch algebraischen Ausdrücke (Implikationen) stellen die Wahrheit der Korrektheitsaussage oben sicher? Gemäß welchen Beweisregeln stellen sie sie sicher?

3. Sind alle fraglichen Implikationen der logischen Konstante wahr äquivalent?

4. Beschreiben Sie kurz in Worten die Funktion eines Programmsegments, das die Vorbedingung V und die Nachbedingung P hat und nur die Variablen i und j verändert. Betrachten Sie dabei nicht das gegebene Programmsegment selbst.

(Ende)

Übung 24: Konstruktion eines Unterprogramms

(Zusammenfügen sortierter Felder)

Konstruieren Sie ein Unterprogramm "Zus2", das die in den zwei sortierten Feldern D(id), id=1, 2, ... nd, und E(ie), ie=1, 2, ... ne, gespeicherten Werte zusammenfügt und sie in das Feld F(if), if=1, 2, ... nd+ne sortiert überträgt. Das Feld D ist **ab**steigend, das Feld E **auf**steigend sortiert. Das Feld F soll **ab**steigend sortiert sein.

Das Unterprogramm Zus2 darf nur die Werte der Feldvariablen F(if), if=1, 2, ... nd+ne verändern, d.h. (call Zus2).d=d bis auf die Werte dieser Feldvariablen.

Der Systemkonstrukteur hat die folgenden Vor- und Nachbedingungen V bzw. P vorgegeben:

V: $nd \in Z$ und $0 \le nd$ und $ne \in Z$ und $0 \le ne$
und$_{i=1}^{nd-1}$ $D(i) \ge D(i+1)$ und$_{i=1}^{ne-1}$ $E(i) \le E(i+1)$ [D ab-, E aufsteigend]
und $\{D(1), ... D(nd), E(1), ... E(ne)\}$ linear angeordnet
und$_{i=1}^{nd+ne}$ $\{D(1), ... D(nd), E(1), ... E(ne)\} \subseteq$ Menge."F(i)" [F vereinbart]

P: $[F(1), ... F(nd+ne)]$ Perm $[D(1), ... D(nd), E(1), ... E(ne)]$
und$_{i=1}^{nd+ne-1}$ $F(i) \ge F(i+1)$ [F absteigend]

Fragen:

1. Was ist eine geeignete Schleifeninvariante? Wie haben Sie sie festgelegt? Geben Sie Ihre Schleifeninvariante sowohl in der Form eines Diagramms als auch in der Form eines logischen (Booleschen) Ausdrucks an.

2. Wie (mit Hilfe welcher Beweisregeln) haben Sie die Initialisierung Ihrer Schleife konstruiert? Die while-Bedingung? Den Kern der Schleife?

3. Geben Sie die an jeder Stelle Ihres Unterprogramms herrschende Zwischenbedingung an. D.h., geben Sie alle für einen Korrektheitsbeweis benötigten Zwischenbedingungen an.

4. Wie haben Sie bereits bei der Konstruktion des Unterprogramms dafür gesorgt, daß seine Korrektheit bewiesen werden kann?

(Ende)

Übung 25: Konstruktion eines Unterprogramms

(Feldsuche)

Konstruieren Sie ein Unterprogramm Fs, das die folgenden Anforderungen erfüllt:

1. {V} call Fs {P} strikt
2. (call Fs).d=[(r2, Z, .), (r1, Z, .)]&d für alle d∈ V

Die Vor- und Nachbedingungen V bzw. P sind:

V: $nd \in Z$ **und** $z \in Z$ **und** $1 \leq z \leq n+1$ **und** $sp \in M$ **und**$_{i=1}^{n} x(i) \in M$

P: $1 \leq z \leq r1 \leq r2 \leq n+1$
und$_{i=z}^{r1-1} x(i)=sp$
und (r1=n+1 oder $x(r1) \neq sp$)
und$_{i=r1}^{r2-1} x(i) \neq sp$
und (r2=n+1 oder x(r2)=sp)

wobei M eine nicht leere, sonst nicht näher spezifizierte Menge ist.

Fragen:

1. Was ist eine geeignete Schleifeninvariante? Wie haben Sie sie festgelegt? Geben Sie Ihre Schleifeninvariante sowohl in der Form eines Diagramms als auch in der Form eines logischen (Booleschen) Ausdrucks an.

2. Wie (mit Hilfe welcher Beweisregeln) haben Sie die Initialisierung Ihrer Schleife konstruiert? Die while-Bedingung? Den Kern der Schleife?

3. Geben Sie die an jeder Stelle Ihres Unterprogramms herrschende Zwischenbedingung an. D.h., geben Sie alle für einen Korrektheitsbeweis benötigten Zwischenbedingungen an.

4. Wie haben Sie bereits bei der Konstruktion des Unterprogramms dafür gesorgt, daß seine Korrektheit bewiesen werden kann?

5. Beweisen Sie die Korrektheit Ihres Unterprogramms.

6. Was macht ein Unterprogramm mit der gegebenen Vorbedingung V und der gegebenen Nachbedingung P?

(Ende)