

Strukturiertes Programmieren

Nachdem wir in der letzten Ausgabe ein einfaches Spiel entwickelt haben, das zum Teil mit Struktogrammen dokumentiert wurde, bringen wir jetzt den Rest des Listings und eine Gegenüberstellung von Flußdiagramm und Struktogramm.

Diese Ausführung des Spiels »TIC TAC TOE« ist recht einfach. Nach gründlicher Überlegung könnte man dieses Programm sofort kodieren. In der Praxis und mit wachsender Programmiererfahrung werden die Aufgaben jedoch bald umfangreicher sein. Professionelle Programmierer fertigen sich, bevor sie anfangen zu kodieren, eine grafische Darstellung des Problemlösungsweges an. Grafiken überblickt der Mensch wesentlich schneller als einfachen Text oder sogar Zahlenkolonnen. Das wohl am häufigsten eingesetzte grafische Hilfsmittel ist der Programmablaufplan (auch Flußdiagramm, Blockdia-

gramm oder im Englischen: Program Flowchart genannt). Es wird jedoch in den letzten Jahren immer mehr von Struktogrammen (auch Nassi-Shneidermann-Diagramm oder NS-Diagramm genannt) abgelöst. Der Hauptgrund war wohl die Entwicklung höherer Programmiersprachen, die eine Strukturierung eines Programms zulassen. Vor allem sind das Pascal, Fortran 77, Cobol, und PL/I. Selbst die neuesten Basic Versionen (zum Beispiel das extended Microsoft-Basic oder, etwas eingeschränkt, das Basic 3.5 der neuen Commodore 16 und 264) besitzen Befehle, die eine Strukturierung eines Programms zulassen. Da-

mit ist vor allem der vollkommene Verzicht von GOTO-Sprunganweisungen gemeint. Im Gegensatz zu Programmablaufplänen (PAP) ist in Struktogrammen eine Darstellung von GOTO-Befehlen nicht vorgesehen und wird somit auch nicht unterstützt. Wer mit normalem Standard-Basic arbeitet, das keine Strukturierungsbefehle wie IF..THEN..ELSE, REPEAT UNTIL, DO WHILE und so weiter kennt, braucht dennoch nicht auf Struktogramme verzichten. Wenn man sich an die im Heft 4/84 beschriebenen Regeln zur Strukturierung hält, ist eine Anwendung von Struktogrammen sogar recht sinnvoll.

Fortsetzung auf Seite 40

```
10 REM *****
20 REM * KREUZ UND QUER *
30 REM *****
90 PRINT "U"
100 DIM SS(9):DIM BW(8)
110 GOSUB1000:REM SPIELFELD INITIALISIEREN
120 GOSUB2000:REM ANZEIGEN
130 GOSUB3000:REM ZUG HOLEN
140 GOSUB2000:REM ANZEIGEN
150 GOSUB4000:REM AUF SPIELLENDE PRUEFEN
160 GOSUB5000:REM COMPUTERZUG
170 GOSUB2000:REM ANZEIGEN
180 GOSUB4000:REM AUF SPIELLENDE PRUEFEN
190 GOTO130
1000 REM-----
1010 REM INITIALISIERUNG
1020 :
1100 FORP=1TO9
1110 :SS(P)=0
1120 NEXTP
1130 RETURN
2000 REM-----
2010 REM ANZEIGEROUTINE
2020 :
2100 FORP=1TO9
2110 :IFSS(P)=1THENPRINT "X";
2120 :IFSS(P)=-1THENPRINT "O";
2130 :IFSS(P)=0THENPRINT ".";
2140 :IFINT(P/3)=P/3THENPRINT
2150 NEXTP
2160 PRINT:PRINT
2170 RETURN
3000 REM-----
3010 REM SPIELERZUG
3020 :
3040 INPUT "GIB ZUG EIN ";P
3050 IFSS(P)<>0THENPRINT "DIESES QUADRAT
IST BESETZT":GOTO3000
3060 SS(P)=1
3070 RETURN
3090 :
```

```
4000 REM-----
4010 REM AUF SPIELLENDE PRUEFEN
4020 :
4040 GOSUB6000
4050 FORP=1TO8
4060 :IFBW(P)=-3THENPRINT "ICH GEWINNE":E
ND
4070 :IFBW(P)=3THENPRINT "DU GEWINNST":E
ND
4080 NEXTP
4090 FORP=1TO9
4100 :IFSS(P)=0THENRETURN
4110 NEXTP
4120 PRINT "UNENTSCHEIDEN":END
4130 :
5000 REM-----
5010 REM COMPUTERZUG
5020 :
5030 GOSUB6000
5040 MT=0
5050 FORP=1TO8
5060 :IFBW(P)=-2THENGOSUB7000:RETURN
5070 :IFBW(P)=2THENMT=MT+1
5080 NEXTP
5090 IFMT=2THENPRINT "AUCH GUT. ANGEGER!"
:END
5100 IFMT=1THENGOSUB8000:RETURN
5110 GOSUB9000
5120 RETURN
5130 :
6000 REM-----
6010 REM SPIELFELD BEWERTEN
6020 :
6030 :
6040 BW(1)=SS(1)+SS(2)+SS(3)
6050 BW(2)=SS(4)+SS(5)+SS(6)
6060 BW(3)=SS(7)+SS(8)+SS(9)
6070 BW(4)=SS(1)+SS(4)+SS(7)
6080 BW(5)=SS(2)+SS(5)+SS(8)
6090 BW(6)=SS(3)+SS(6)+SS(9)
6100 BW(7)=SS(1)+SS(5)+SS(9)
6110 BW(8)=SS(3)+SS(5)+SS(7)
```

```
6120 RETURN
6130 :
7000 REM-----
7010 REM SIEGZUG
7020 :
7040 GOSUB9000
7050 GOSUB6000
7060 FORP=1TO8
7070 :IFBW(P)=-3THENRETURN
7080 NEXTP
7090 GOSUB10000
7100 GOTO7000
7110 :
7120 :
8000 REM-----
8010 REM SPERREN
8030 :
8040 GOSUB9000
8050 GOSUB6000
8060 F=0
8070 FORP=1TO8
8080 :IFBW(P)=2THENF=1
8090 NEXTP
8100 IFF=0THENRETURN
8110 GOSUB10000
8120 GOTO8000
8130 :
9000 REM-----
9010 REM ZUFALLSZUG
9020 :
9040 CM=9*RND(0)+1
9050 IFSS(CM)<>0THEN9000
9060 SS(CM)=-1
9070 RETURN
9080 :
9090 :
10000 REM-----
10010 REM ZURUECKNEHMEN
10030 :
10040 SS(CM)=0
10050 RETURN
```

Listing. Das im letzten Heft entwickelte Spiel »TIC TAC TOE« als komplettes Listing. Verbessern Sie die Grafik und die Strategie.

Doch zunächst möchte ich Ihnen die Symbolik des PAP vorstellen. Schauen wir uns die Tabelle an. Hier sind die wichtigsten und gebräuchlichsten Elemente des PAP aufgeführt.

Symbol 1: Bearbeitung

Man trägt Anweisungen ein; die Schreibweise der Eintragung orientiert sich oft an der zu verwendenden Programmiersprache. Es kann auch eine Gruppe von Anweisungen eingetragen werden.

Symbol 2: Verzweigung

Wenn in einem Programm in Abhängigkeit einer Bedingung der lineare Ablauf unterbrochen werden soll, erfolgt die Eintragung der Bedingung in diesem Symbol.

Symbol 3: Unterprogrammaufruf

Hier wird die Bezeichnung eines Unterprogramms eingetragen. In Basic setzt man am Besten noch die Anfangszeilennummer dazu.

Symbol 4: Programmmodifikation

Hier trägt man eine Aktion ein, die den Ablauf an einer zeitlich dahinterliegenden Stelle ändert (wird selten gebraucht).

Symbol 5: Operation von Hand

In manchen Programmen kommt es vor, daß an einer bestimmten Stelle die Diskette oder Kassette gewechselt werden muß. Diese Aktion wird hier eingetragen.

Symbol 6: Eingabe, Ausgabe

Man benutzt dieses Symbol immer, wenn eine Eingabe- oder Ausgabeanweisung im Programmablauf auftritt. Man trägt dabei die Art der Ein-/Ausgabe ein (INPUT oder PRINT oder ähnliches) und den Datensatz, der gelesen oder geschrieben wird.

Symbol 7: Flußlinie

Sie verbindet die Symbole miteinander. Die Pfeilspitzen können weggelassen werden, wenn die Flußrichtung eindeutig ist (normalerweise immer von oben nach unten, von links nach rechts).

Symbol 8: Konnektor, Übergangsstelle

Die Eintragung ist beliebig. Dieses Symbol wird benutzt, um nicht durch zu viele Flußlinien die Übersicht zu verlieren.

Symbol 9: Grenzstelle, Anschlußmarke

Dieses Symbol enthält eine Bezeichnung von Programmbeginn und -ende sowohl von Haupt- als auch von Unterprogrammen.

Symbol 10: Bemerkung

Falls der Platz innerhalb eines Symbols nicht ausreicht, um es zu beschriften, kann man mit diesem Symbol eine Bemerkung hinzufügen.

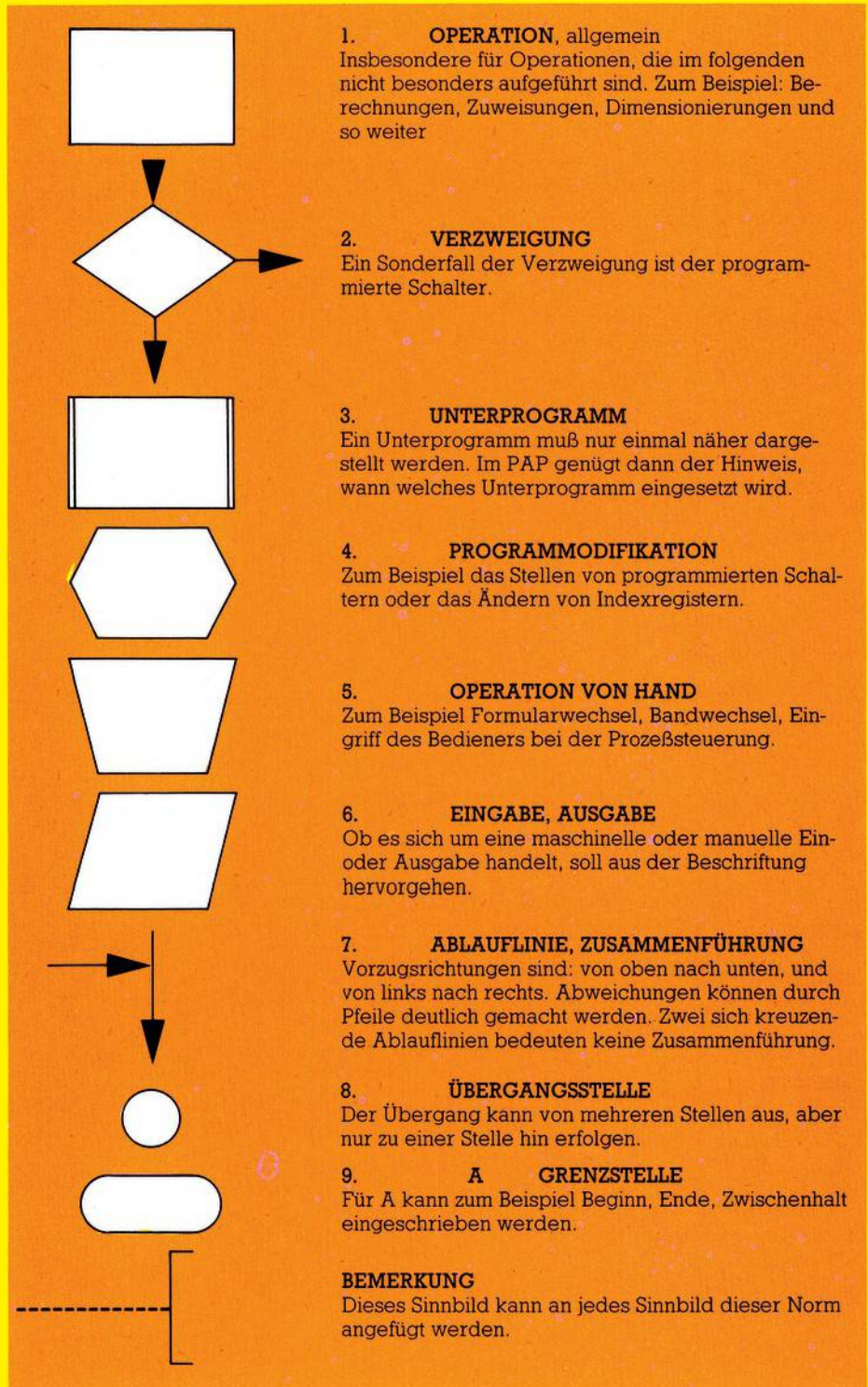


Tabelle 1: Sinnbilder für Programmablaufpläne

Unternehmen gehen bei Programmiervorhaben immer mehr dazu über, anstatt von Flußdiagrammen Struktogramme einzusetzen. Sie zwingen den Programmierer zu einer übersichtlichen Programmierung und erlauben es ihm nicht, durch viele GOTOs das Programm »undurchsichtig« zu machen. Wenn man Struktogramme verwendet, kann man auf Programmablaufpläne völlig verzichten. Als Programmierer sollte man beide Darstellungsarten kennen, denn Programmablaufpläne und Strukto-

gramme werden noch lange nebeneinander bestehen.

Die Elemente des Struktogramms

Jede Aufgabenstellung läßt sich mit den folgenden Grundstrukturen lösen:

Sequenz Auswahl Wiederholung

Diese Grundstrukturen werden zu Strukturblocken zusammengesetzt und sind beliebig ineinander verschachtelbar.

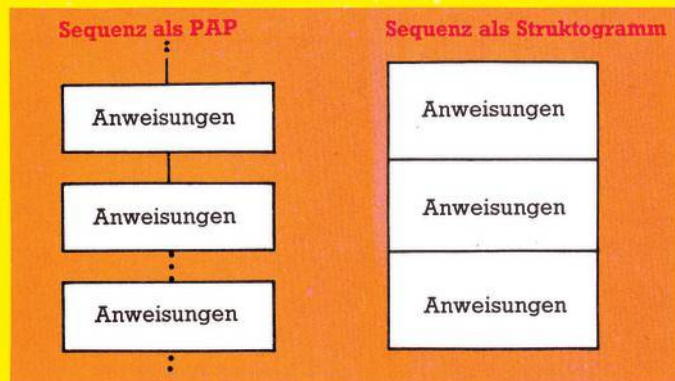


Bild 1. Die Sequenz ▲

Bild 3. Die Anweisungen 1 werden nur dann ausgeführt, wenn die Bedingung B erfüllt ist, sonst werden die Anweisungen 2 ausgeführt.

Bild 6. Die Anweisungen werden ausgeführt, und wenn die Wiederholungsbedingung erfüllt ist, werden die Anweisungen wieder ausgeführt, sonst wird die Schleife verlassen. Um eine Endlosschleife zu vermeiden, muß auch hier die Wiederholungsbedingung innerhalb der Gruppe der Anweisungen verändert werden. ▼

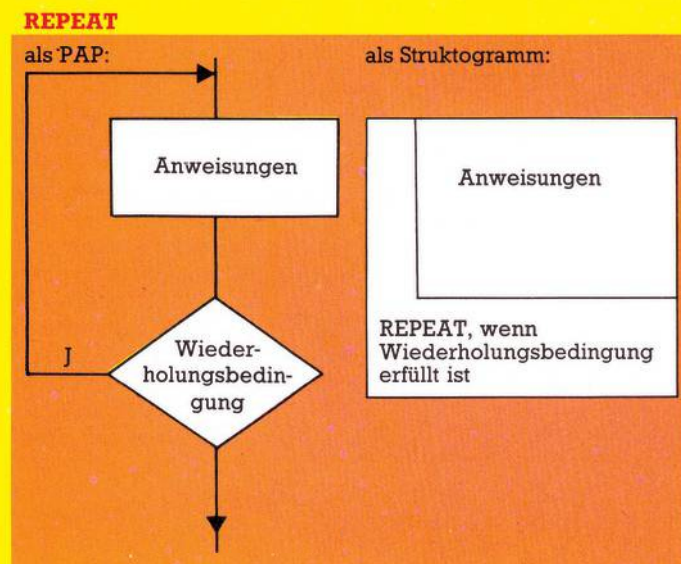


Bild 4. Die Anweisungen 1 werden nur dann ausgeführt, wenn die Variable den Wert 1 enthält, die Anweisungen 2 werden nur dann ausgeführt, wenn die Variable den Wert 2 enthält, die Anweisungen n werden nur dann ausgeführt, wenn die Variable den Wert n enthält. Es können beliebig viele Blöcke von Anweisungen vorkommen. ►

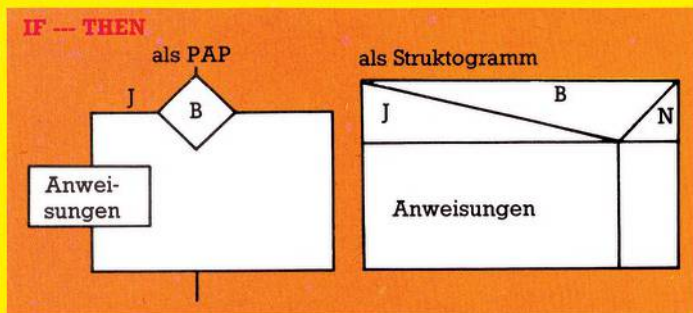
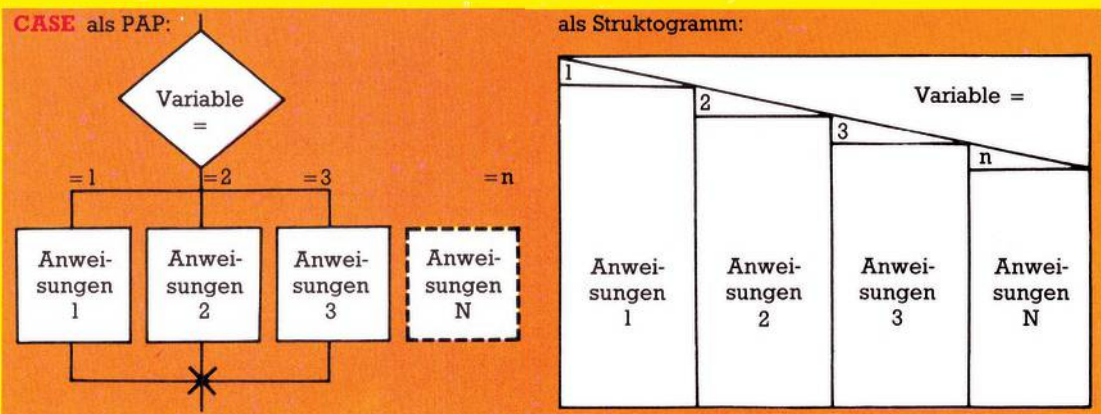


Bild 2. Die Anweisungen werden nur dann ausgeführt, wenn die Bedingung B erfüllt ist. ▲

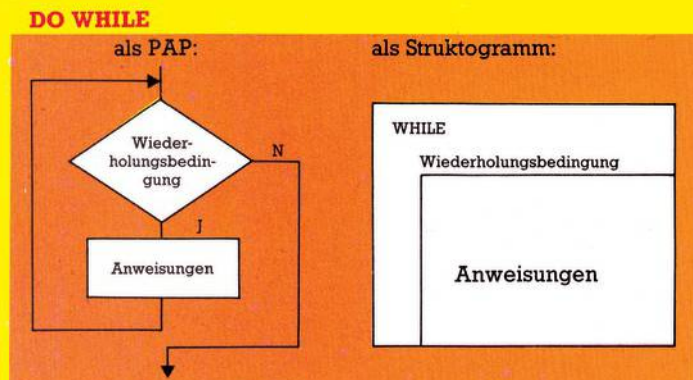
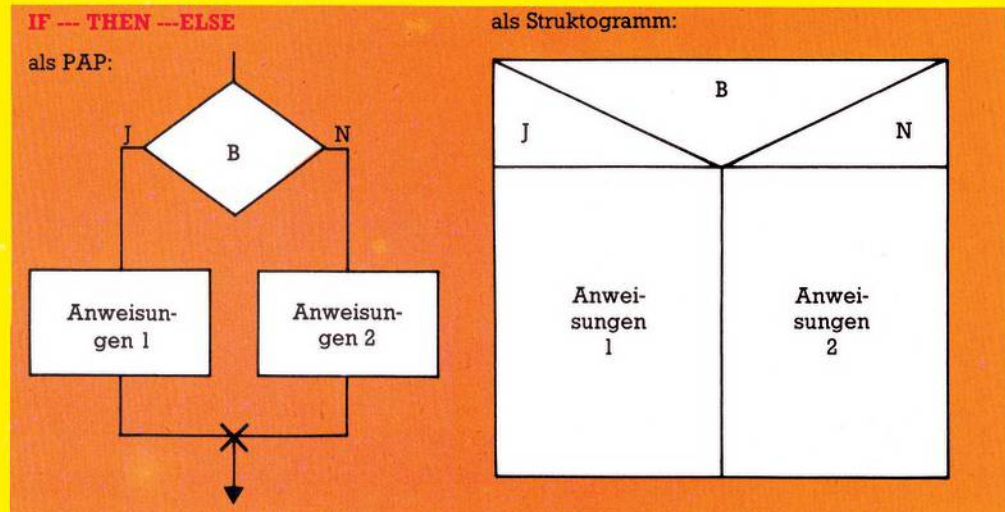


Bild 5. Die Anweisungen werden ausgeführt, solange (=while) die Wiederholungsbedingung erfüllt ist. Dabei wird jedesmal vor Beginn überprüft. Um eine Endlosschleife zu vermeiden, ist es erforderlich, daß die Wiederholungsbedingung innerhalb der Gruppe der Anweisungen verändert wird. ▲

Strukturiertes Programmieren

Sequenz

Eine Sequenz liegt vor, wenn mehrere Anweisungen hintereinander ausgeführt werden sollen (Bild 1).

In die jeweiligen Kästchen trägt man jeweils einen Befehl ein oder auch die Bezeichnung kompletter Programmteile.

Auswahl

Man unterscheidet drei Formen: **IF.THEN / IF.THEN..ELSE / CASE**

Man benutzt diese Formen, um Befehle oder Befehlsgruppen oder auch komplette Programmteile nur dann ausführen zu lassen, wenn eine Bedingung erfüllt ist (Bild 2, 3 und 4).

Wiederholung

Man benutzt zwei Formen der Wiederholung:

DO WHILE / REPEAT

Man benutzt diese Formen, um Befehle oder Befehlsgruppen oder auch komplette Programmteile mehrmals hintereinander ausführen zu lassen. Eine Programmschleife kann nur mit diesen Formen realisiert werden (Bild 5 und 6).

Facit

Beim Struktogramm entfällt völlig die Flußlinie, sie wird ersetzt durch die gemeinsame waagerechte Begrenzungslinie. Struktogramme kann man immer in Programmablaufpläne übertragen, umgekehrt ist es oft nur unter erheblichem Aufwand möglich. Das bedeutet, daß man durchaus auch mit Flußdiagrammen strukturiert programmieren kann; der Zwang dazu entfällt jedoch. Bei der Planung seiner Programme mit Struktogrammen wird man es schwer haben, »Spaghetti-Kode« zu produzieren. Und das ist allemal ein Vorteil. Und machen Sie bitte nicht den Fehler wie so viele (ich schließe mich dabei nicht aus) und erstellen Sie Ihr Flußdiagramm oder Struktogramm erst dann, nachdem Sie kodiert haben und Sie durch Ihr eigenes Programm nicht mehr durchsteigen. (gk)

