

Darstellungsformen für Algorithmen

Es gibt verschiedene **Darstellungsformen**, um algorithmische Grundbausteine und Algorithmen darzustellen. An dieser Stelle soll auf die zwei wichtigsten eingegangen werden:

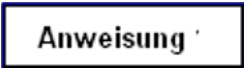

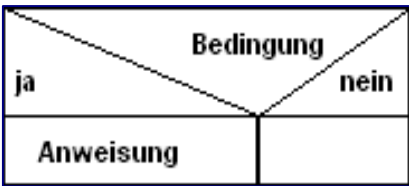
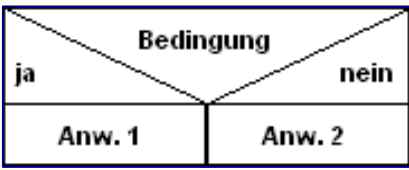
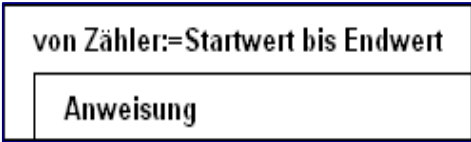
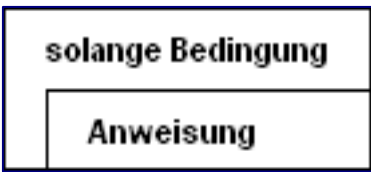
Pseudocode

Der Pseudocode stellt einen Algorithmus in einer Schreibweise in der Art einer Programmiersprache dar, die aber in der Regel näher an der natürlichen Sprache ist als am formalen Programmcode. Es gibt keine Standardisierung bzw. Vorschrift, wie ein Pseudocode auszusehen hat; die Ablaufstrukturen werden grob durch Texte und fest vorgeschriebene Schlüsselwörter beschrieben. Guter Pseudocode zeichnet sich durch eine kurze präzise Beschreibung des Algorithmus aus.

Struktogramm

Struktogramme (auch nach ihren Entwicklern **Nassi-Shneidermann-Diagramme** genannt) ist eine genormte (DIN 66261) Entwurfsmethode für Algorithmen. Die Methode zerlegt ein Gesamtproblem, das man mit dem gewünschten Algorithmus lösen will, in immer kleinere Teilprobleme bis schließlich nur noch elementare Grundstrukturen wie Anweisungsfolgen (Sequenzen) und Kontrollstrukturen zur Lösung des Problems übrigbleiben. In der Softwareentwicklung werden Struktogramme sehr selten eingesetzt, da normaler Programmcode einfacher zu schreiben und zu verändern ist: Korrigiert man einen Fehler oder macht eine Ergänzung, muss man ein Struktogramm in der Regel komplett neu zeichnen. Im **Informatik-Unterricht** werden Struktogramme dagegen verwendet, damit Schüler den Aufbau logischer Abläufe, die für die Programmierung nötig sind, trainieren können. Die Umsetzung von Struktogrammen in Programmcode ist immer noch Bestandteil der schulischen Ausbildung.

Grundelemente des Pseudocode und von Struktogrammen mit Python-Beispielen (Auswahl)

Strukturelement	Pseudocode	Struktogramm	Python-Beispiel
Verarbeitung (Elementarblock)	Anweisung		<code>anzahl = 17</code>
Reihenfolge (Sequenz)	Anweisung1 Anweisung2 Anweisung3		<code>zahl = 11</code> <code>ergebnis = zahl * zahl</code> <code>print ergebnis</code>
Verzweigung (ohne Alternative)	<u>wenn</u> Bedingung <u>dann</u> Anweisung		<code>if anzahl > 167:</code> <code> anzahl = anzahl / 2</code>
	<u>wenn</u> Bedingung <u>dann</u> Anweisung1 <u>sonst</u> Anweisung2		<code>if bmi < 16:</code> <code> print "untergewichtig"</code> <code>else:</code> <code> print "Normalgewicht"</code>
Zählschleife	<u>von</u> Zähler:=Startwert <u>bis</u> Endwert Anweisung		<code>for zaehler in range (1,11):</code> <code> print zaehler</code>
anfangsgeprüfte Schleife	<u>solange</u> Bedingung Anweisung		<code>zaehler = 0</code> <code>while zaehler < 11:</code> <code> print zaehler</code> <code> zaehler = zaehler + 1</code>

Ausblick: Das Konzept für das Nassi-Shneiderman-Diagramm (Struktogramm) stammt aus den 1960er-Jahren und bildet einen linearen Programmfluss ab. Dies ist insbesondere für **imperative** Programmiersprachen gut geeignet. Für die Abbildung **objektorientierter** Programmkonzepte hat sich dieses Konzept als ungeeignet erwiesen. Als Konsequenz wurde die Unified Modelling Language (UML) für objektorientierte Programmkonzepte entwickelt.