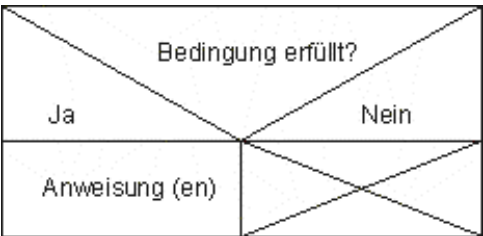
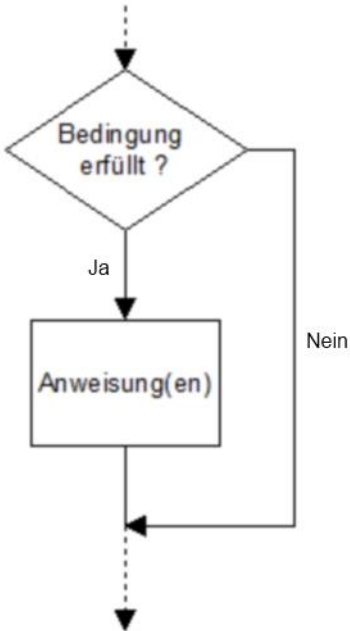


Einseitige Auswahl

Das wesentliche Kennzeichen der einseitigen Auswahl ist, dass in Abhängigkeit vom Wahrheitsgehalt einer Bedingung im Ja-Zweig der Auswahlstruktur eine oder mehrere Anweisungen ausgeführt werden. Der Nein-Zweig bleibt bei der einseitigen Auswahl leer. Im Anschluss an diese Programmstruktur wird der Programmablauf wieder zusammengeführt.

Darstellung der einseitigen Auswahl im Struktogramm bzw. Programmablaufplan

Struktogramm nach DIN 66261	Bemerkungen	PAP DIN 66001
	<p>Bei der einseitigen Auswahlstruktur wird in Abhängigkeit von einer Bedingung eine Anweisung bzw. ein Anweisungsblock durchlaufen.</p> <p>Im anderen Fall ist keine Anweisung vorgesehen.</p>	

Syntax der einseitigen Auswahl in Python:

Syntax 1	<pre>if Bedingung: Anweisung #keine Einrückung ...</pre>
Syntax 2	<pre>if Bedingung: Anweisung 1 Anweisung 2 ... #Einrückung zurück ...</pre>

Wenn im Ja-Zweig der einseitigen Auswahl nur eine Anweisung notwendig ist, kann auf die Verwendung der Einrückung verzichtet werden (Syntax 1).

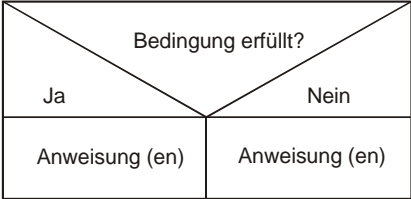
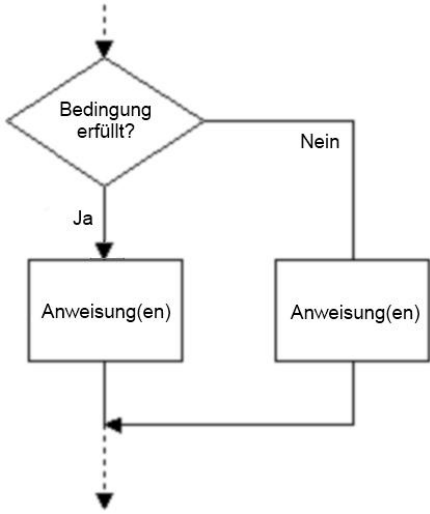
Aus Gründen der Übersichtlichkeit werden alle zum Ja-Zweig gehörenden Anweisungen um eine Tabulatorposition eingerückt.

Die Codierung der einseitigen Auswahl beginnt mit dem Schlüsselwort **if** gefolgt von der Bedingung.

Zweiseitige Auswahl

Bei der zweiseitigen Auswahl stehen sowohl im Ja-Zweig als auch im Nein-Zweig der Auswahlstruktur eine oder mehrere Anweisungen, die in Abhängigkeit vom Wahrheitsgehalt der Bedingungsprüfung ausgeführt werden.

Darstellung der einseitigen Auswahl im Struktogramm bzw. Programmablaufplan

Struktogramm nach DIN 66261	Bemerkungen	PAP DIN 66001
	<p>Die zweiseitige Auswahlstruktur ermöglicht es, in Abhängigkeit von einer Bedingung zwischen zwei Anweisungen (bzw. Anweisungsblöcken) zu wählen.</p> <p>Je nach Zutreffen der Bedingung wird der Ja- oder Nein-Zweig der Auswahl durchlaufen.</p>	

Syntax der zweiseitigen Auswahl in Python:

Syntax	<pre> if Bedingung: Anweisung 1 Anweisung 2 ... else: Anweisung 1 Anweisung 2 ... </pre>
---------------	--

Wenn im Ja-Zweig oder im Nein-Zweig der zweiseitigen Auswahl nur eine Anweisung notwendig ist, kann auf die Verwendung der geschweiften Klammern verzichtet werden.

Aus Gründen der Übersichtlichkeit werden auch bei der zweiseitigen Auswahl die Anweisungen der Zweige eingerückt.

Die Codierung der zweiseitigen Auswahl beginnt mit dem Schlüsselwort **if** gefolgt von der Bedingung in runden Klammern. Es folgen in den geschweiften Klammern die Anweisungen des Ja-Zweiges. Der Nein-Zweig beginnt mit dem Schlüsselwort **else**, gefolgt von dem Anweisungsblock des Nein-Zweiges.

Mehrseitige Auswahl

Bei der mehrseitigen Auswahl werden mehrere Auswahlstrukturen ineinander verschachtelt. Für die einzelnen Auswahlstrukturen gelten die Erläuterungen zur ein- bzw. zweiseitigen Auswahlstruktur. Obwohl man prinzipiell Auswahlstrukturen beliebig tief schachteln kann, sollte man aus Gründen der Übersichtlichkeit nicht mehr als zwei oder drei Auswahlstrukturen ineinander verschachteln.

Darstellung der einseitigen Auswahl im Struktogramm bzw. Programmablaufplan

Struktogramm nach DIN 66261	Bemerkungen	PAP DIN 66001
	<p>Bei der Mehrfachauswahl sind Auswahlstrukturen geschachtelt angeordnet.</p> <p><i>(Diagramme beschreiben nicht den gleichen Sachverhalt!)</i></p>	

Syntax der mehrseitigen Auswahl in Python:

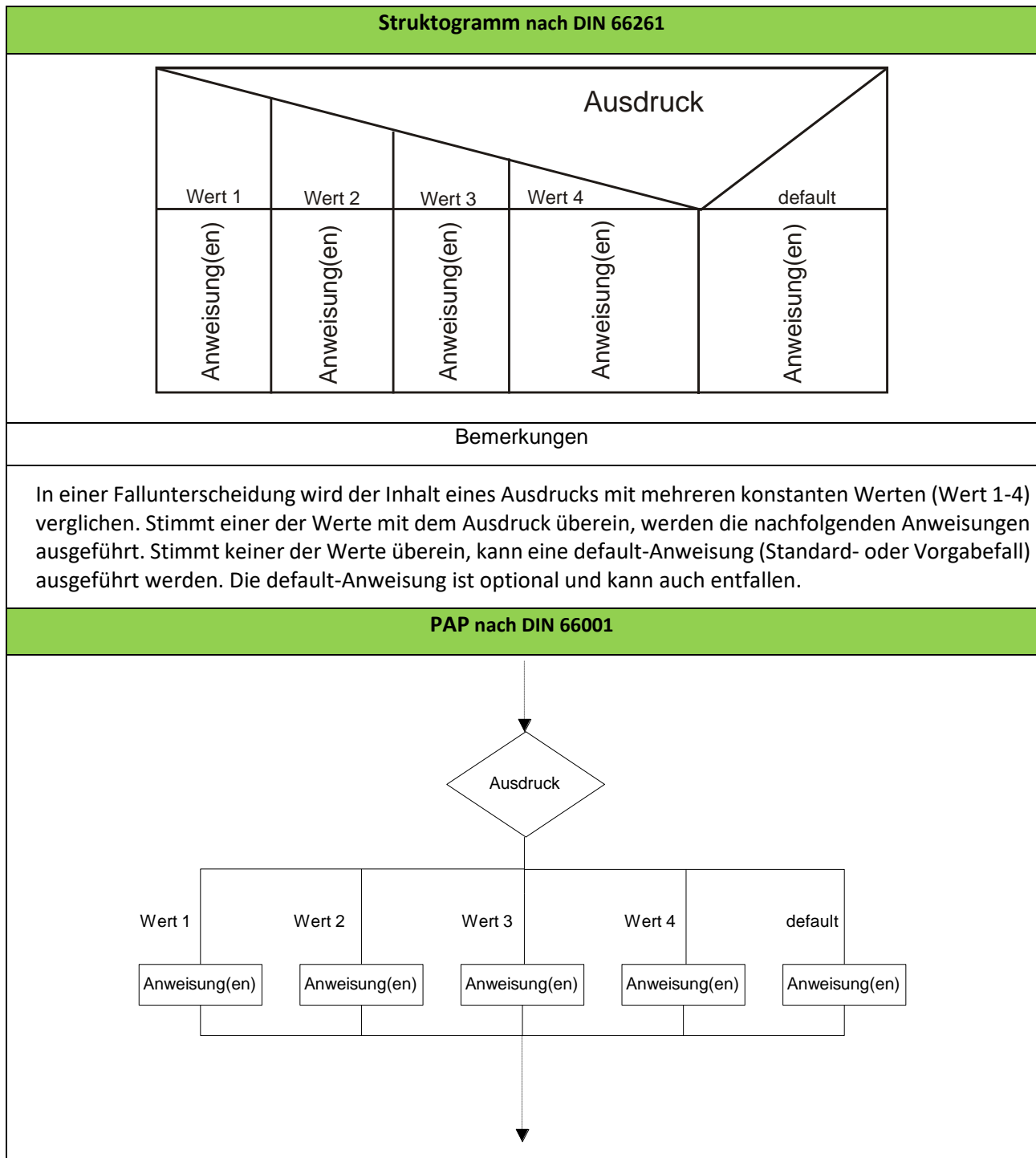
Syntax	<pre> if Bedingung 1: Anweisung ... else: if Bedingung 2: Anweisung ... else: Anweisung ... #Einrückungen zurück ... </pre>
---------------	--

Bei der mehrseitigen Auswahl ist der konsequente Einsatz von Einrückung besonders wichtig, da der Quelltext ansonsten unverständlich wird.

Fallunterscheidung

Die Fallunterscheidung stellt eine vereinfachte Form der Mehrfachauswahl dar. Die Fallunterscheidung erhöht vor allem dann die Übersichtlichkeit eines Programms, wenn viele Auswahlalternativen zur Verfügung stehen und bietet sich insbesondere für die Steuerung eines Programms über Auswahlmenüs an.

Darstellung der Fallunterscheidung im Struktogramm bzw. PAP:



Syntax der Fallunterscheidung in Python:

Syntax

```
if Ausdruck==Wert1:
    Anweisung
...
elif Ausdruck==Wert2:
    Anweisung
...
elif Ausdruck==Wert3:
    Anweisung
...
elif Ausdruck==Wert4:
    Anweisung
...
else:    # default
    Anweisung
...

#Einrückungen zurück
...
```

Eine Auswahlstruktur mit konstantem Ausdruck und eine Fallunterscheidung bzgl. Konstanter Werte werden in Python nicht durch ein besonderes Sprachkonstrukt unterstützt.

Vielmehr muss diese Auswahlstruktur durch eine flache „Mehrfachverzweigung“ realisiert werden.

Dabei bleibt der Vergleichsausdruck konstant und wird mit den möglichen (erlaubten) Werten verglichen in jeweils einer Verzweigung (if, elif).

Der „default-Fall“ wird durch den abschließenden else-Pfad realisiert.