9 Alternativen

Lernziele In diesem Abschnitt lernst du:

- Bei einer `if`-Bedingung auch eine Alternative anzugeben.

Einführung Mit `if` kontrollierst du, ob ein bestimmtes Stück in deinem Programmcode ausgeführt werden soll oder nicht. Das haben wir im letzten Abschnitt verwendet, um zu unterscheiden, ob eine Zahl positiv oder negativ ist. Für eine solche Unterscheidung von *zwei Alternativen* gibt es in Python eine Kurform: Das `else`.

`else` heisst übersetzt «andernfalls» und ersetzt das zweite `if`. Die beiden Codebeispiele hier sind absolut gleichwertig:

```
if zahl >= 0:
    print sqrt(zahl)
if zahl < 0:
    print "Zahl negativ"
```

```
if zahl >= 0:
    print sqrt(zahl)
else:
    print "Zahl negativ"
```

Allerdings hat das `else` natürlich nur deshalb Sinn, weil sich die beiden `if` links ergänzen: Entweder ist `zahl >= 0` erfüllt oder dann `zahl < 0`.

Neben der einfachen `if`-Bedingung gibt es also noch eine `if-else`-Unterscheidung zwischen zwei Alternativen. Das `else` selber hat nie eine eigene Bedingung, sondern tritt immer dann in Kraft, wenn die Bedingung im `if` zuvor *nicht* erfüllt war.

Das Programm Das Osterdatum ändert sich jedes Jahr und muss daher immer neu berechnet werden. Carl Fiedrich Gauss hat für diese Berechnung ein Verfahren (Algorithmus) vorgestellt. Weil Ostern immer im März oder April sind, gibt seine Formel den Tag ab dem 1. März an. Der Tag «32» entspricht dann einfach dem 1. April.

In unserem Programm berechnet `easterday` aus dem Modul `tjaddons` den Ostertag. Danach müssen wir aber selber noch prüfen, ob es im März oder April liegt. Weil wir nur zwei Alternativen haben, können wir das mit `if-else` machen (Zeilen 5 und 11). In der Zeile 6 berücksichtigen wir noch eine Spezialregelung: Wenn das Datum auf den 26. April fällt, dann wird Ostern auf den 19. April vorverschoben.

```
1 from tjaddons import *
2
3 Jahr = inputInt("Gib ein Jahr ein:")
4 Tag = easterday(Jahr)
```

© 2017, Tobias Kohn

```
5  if Tag > 31:
6      if Tag == 57:
7          Tag = 19
8      else:
9          Tag -= 31
10     msgDlg(Tag, "April", Jahr)
11 else:
12     msgDlg(Tag, "März", Jahr)
```

In diesem Programm kommen zwei `else` vor. Dasjenige in der Zeile 8 gehört zum `if` in der Zeile 6 und das in der Zeile 11 gehört zum `if` in der Zeile 5. Woher weiss der Computer, welches `else` zu welchem `if` gehört? An der Einrückungstiefe! Jedes `else` muss genau gleich eingerückt sein wie das `if`, zu dem es gehört.

Die wichtigsten Punkte Bedingungen mit `if` können entweder alleine stehen oder zusammen mit einer Alternative. Diese wird über `else` eingeleitet und wird immer dann ausgeführt, wenn die Bedingung bei `if` nicht erfüllt ist und der Computer den Code von `if` überspringt.

```
if Bedingung:
    Code ausführen, wenn die
    Bedingung erfüllt ist.
else:
    Code ausführen, wenn die
    Bedingung nicht erfüllt ist.
```

AUFGABEN

37. Bei der Collatz-Vermutung startest du mit einer Zahl x und erzeugst dann eine Zahlenfolge noch dem folgenden Prinzip: Wenn x gerade ist, dann teile durch 2. Andernfalls multipliziere x mit 3 und zähle 1 dazu:

$$17 \quad 52 \quad 26 \quad 13 \quad 40 \quad 20 \quad 10 \quad 5 \quad 16 \quad 8 \quad 4 \quad 2 \quad 1$$

Schreibe ein Programm, das zu einer beliebigen Startzahl 10 Folgezahlen nach diesem Prinzip berechnet und ausgibt.

38.* Du kannst die Osterformel von Gauss auch selbst programmieren. Hier sind die Formeln dafür (natürlich kannst du in Python nicht alles so kompakt schreiben wie hier):

```
k = Jahr // 100,   p = k // 3,   q = k // 4,
M = (15 + k - p - q) % 30,   N = (4 + k - q) % 7
a = Jahr % 19,   b = Jahr % 4,   c = Jahr % 7,
d = (19a + M) % 30,   e = (2b + 4c + 6d + N) % 7
Ostertag = 22 + d + e
```

© 2017, Tobias Kohn