1.7 Schleifen

Definition: Schleife

Eine **Schleife** ist ein Programmbaustein, der eine Anweisungsfolge mehrfach ausführt. Wir unterscheiden drei Arten:

- **Zählschleife** (`for`) — die Zahl der Durchläufe steht vorher fest.
- **Kopfgesteuerte Schleife** (`while`/`while do`) — vor jedem Durchlauf wird geprüft; sie kann *keinmal* laufen.
- **Fußgesteuerte Schleife** (`repeat until`) — erst nach dem Durchlauf wird geprüft; sie läuft **mindestens einmal**.

Python kennt die fußgesteuerte Schleife nicht; man baut sie mit `while True:` und `break` nach.

Beispiel: Summe von a bis b

Python

```python
def summe(a, b):
    ergebnis = 0
    for zahl in range(a, b + 1):
        ergebnis = ergebnis + zahl
    return ergebnis
```

Object Pascal

```
function Summe(a, b: Integer): Integer;
var
  zahl, ergebnis: Integer;
begin
  ergebnis := 0;
  for zahl := a to b do
    ergebnis := ergebnis + zahl;
  Result := ergebnis;
end;
```

Für $a = 5$ und $b = 10$ ergibt sich $5 + 6 + 7 + 8 + 9 + 10 = 45$.

Stolperstein

Python gibt einen Wert mit `return` zurück — und beendet die Funktion damit sofort. Pascal hat kein `return`: Dort wird dem Ergebnis ein Wert **zugewiesen**, und die Funktion läuft weiter bis zum `end`. Dafür gibt es in Object Pascal die vordefinierte Variable `Result`:

```
  Result := ergebnis;
```

Klassisches Pascal erlaubt stattdessen auch die Zuweisung an den Funktionsnamen (`Summe := ergebnis;`). Beides ist in Delphi und Lazarus erlaubt; in diesem Skript steht durchgehend `Result`, weil es in allen Übersetzermodi funktioniert und beim Lesen klarer zeigt, dass hier ein Rückgabewert gesetzt und nicht die Funktion erneut aufgerufen wird. Soll die Funktion sofort verlassen werden, dient dazu `Exit`.

Stolperstein

In Python läuft `range(a, b)` **bis ausschließlich** `b`. In Pascal läuft `for zahl := a to b` **bis einschließlich** `b`. Daher das `+ 1` im Python-Beispiel. Das ist die häufigste Ursache für einen Durchlauf zu wenig.

1.7.1 Welche Schleife wann?

Situation	passende Schleife
„zehnmal" oder „für jedes Element"	`for`
„solange noch nicht fertig", evtl. nie	`while`
„bis der Benutzer `q` drückt" — mindestens einmal	`repeat until`

Die Gefahr liegt immer bei der Abbruchbedingung: Wird sie nie wahr, entsteht eine **Endlosschleife** und das Programm reagiert nicht mehr.

Arbeitsauftrag 3

8. Berechnen Sie die Fakultät $n! = 1 \cdot 2 \cdot \ldots \cdot n$ mit einer Schleife, in beiden Sprachen.
9. Ein Kapital k wird jährlich mit p Prozent verzinst. Geben Sie den Kontostand für jedes Jahr aus, bis er sich verdoppelt hat. Welche Schleifenart passt — und warum nicht die `for`-Schleife?
10. Schreiben Sie eine Schleife, die absichtlich nicht terminiert, und erklären Sie in einem Satz, welche Grundeigenschaft aus Abschnitt 1.3 sie verletzt.

Gymnasium Ottweiler