PROFESSOR ERKLÄRBÄR

Serie Informatik · Folge 9 · Grundlagen

INFORMATIK

LEHRERINFO ZUM VIDEO

Funktionen und Listen

FACH	STUFE	DAUER	VIDEO
Informatik	Klasse 7–9	12 min	youtube.com/@Erklaerbaer

01 Lernziele

Nach dem Video können die Schülerinnen und Schüler …

- eigene Funktionen mit `def` definieren und aufrufen,
- Parameter übergeben und Ergebnisse mit `return` zurückgeben – und `return` von `print` unterscheiden,
- erklären, warum Funktionen Programme kürzer, lesbarer und leichter änderbar machen, und was «lokal» bedeutet,
- Listen anlegen, Elemente per Index (ab 0) lesen sowie mit `append`, `insert` und `remove` ändern,
- Listen mit `for` durchlaufen und auswerten (Summe, Maximum, Durchschnitt),
- eingebaute Funktionen nutzen (`len`, `sum`, `max`, `min`, `sorted`).

02 Voraussetzungen

Variablen und Datentypen (Folge 7); Vergleiche, `if` und Schleifen mit `for` und `while` (Folge 8); Prozentrechnen für den Rabatt-Rechner (Mathe, Folge 2).

03 Ablauf des Videos

ZEIT	KAPITEL	INHALT UND BEISPIEL
0:00	Einstieg	200 Songs – 200 Variablen? · Liste und Funktion · Ident · Fahrplan
0:35	Kopierter Code nervt	Begrüssung für Lea, Ben und Mia dreimal kopiert · eine Kopie vergessen
1:08	Funktionen definieren	Rezept im Kochbuch · `def greet(name):` · Aufrufe mit Konsole · eine Stelle ändern
2:06	Parameter und return	`area(4, 5)` = 20, zwei Zimmer 29 m² · print gegen return · Rabatt 80 € − 25 % PAUSE-FRAGE `double(double(3))` · Merksatz

3:29	Lokale Variablen	`result` ausserhalb → NameError · zwei Funktionen, je ein eigenes `result`
4:00	Listen	Playlist · Index ab 0, `songs[-1]`, `len` · append, insert, remove PAUSE-FRAGE `nums[2]` · Merksatz
5:19	Listen durchlaufen	`for song in songs` · Wochentemperaturen: Summe 115, Ø ≈ 16,4 °C, Maximum 21 · Merksatz
6:17	Funktionen und Listen zusammen	`average(values)` für Temperaturen, Noten, Laufzeiten · Highscores sortieren, Top 3
6:58	Eingebaute Helfer	`len`, `sum`, `max`, `min`, `sorted` · `random.choice` · Bibliotheken
7:36	Projekt-Baustein	Taschenrechner 0.3: vier Rechenfunktionen und Verlauf als Liste
7:59	Typische Fehler	vier Fehlvorstellungen (siehe 04)
8:45	Zusammenfassung	Kernaussagen · eine Liste und eine Schleife statt 200 Variablen
9:22	Quiz	4 Fragen · Endkarte Folge 10

04 Typische Fehlvorstellungen

- Das erste Element hat den Index 1 – Python zählt ab 0; das letzte Element hat den Index `len - 1`.
- `print` in der Funktion gibt das Ergebnis zurück – nur `return` liefert einen Wert; ohne `return` kommt `None` zurück (Aufgabe 3).
- Eine Funktion definieren heisst, sie auszuführen – `def` legt sie nur an; sie läuft erst beim Aufruf.
- `songs[len(songs)]` ist das letzte Element – das ergibt einen IndexError; richtig ist `songs[-1]` oder `songs[len(songs) - 1]`.

05 Einsatz im Unterricht

Einstieg: Die Klasse schreibt die Begrüssung für drei Namen von Hand ab und ändert sie dann – danach Kapitel 1 und 2 schauen.

Listen zum Anfassen: Fünf Lernende stehen als Liste in einer Reihe (Index auf Karten); append, insert und remove werden nachgespielt.

Schreibtischtest: Aufgabe 5 von Hand mit einer Variablentabelle, dann in Thonny prüfen.

Mini-Projekt: Aufgabe 6 (Notenrechner) eignet sich als Hausaufgabe oder Partnerarbeit.

Hinweis: Pythons Listen sind intern dynamische Arrays; das ist Thema der Advanced-Folge 9. Die Liedtitel im Video sind erfunden. `print(total / 7)` zeigt 16.428571428571427 – im Video gerundet als 16,4 °C. Gleitkommazahlen können kleine Rundungsfehler zeigen (z. B. 9.799999999999999 statt 9.8).

06 Quellen

- Python Software Foundation: The Python Tutorial, Kapitel 4.8 «Defining Functions» und 5.1 «More on Lists» (docs.python.org/3/tutorial)
- Python-Dokumentation: Built-in Functions (len, sum, max, min, sorted) · random.choice
- Thonny – Python IDE for beginners, Universität Tartu (thonny.org)
- Lehrplan 21, Medien und Informatik: Bereich «Informatik – Programmieren» (MI.2.2)

NAME _____ KLASSE _____ DATUM _____

Funktionen und Listen

1 Index lesen ●○○

Es gilt `nums = [7, 3, 9, 1, 5]`. **Bestimme**: a) `nums[0]` b) `nums[3]` c) `nums[-1]` d) `len(nums)` e) Was passiert bei `nums[5]`?

2 Umfang berechnen ●○○

Schreibe eine Funktion `perimeter(width, height)`, die den Umfang eines Rechtecks zurückgibt. **Teste** sie mit 4 und 5.

3 print oder return? ●●○

`def double(x): print(2 * x) y = double(4) print(y + 1)` – Was gibt das Programm aus, und wo bricht es ab? **Erkläre** und **korrigiere** die Funktion.

4 Liste verändern ●●○

`fruits = ["Apfel", "Birne"] fruits.append("Kiwi") fruits.insert(0, "Banane") fruits.remove("Birne")` – Wie sieht die Liste nach jedem Schritt aus?

5 Schreibtischtest ●●○

`def count_big(values): n = 0 for v in values: if v > 10: n = n + 1 return n` – Was liefert `count_big([4, 12, 15, 8, 20])`? Notiere v und n nach jedem Durchlauf.

6 Notenrechner

Schreibe ein Programm mit einer Liste deiner letzten Noten und einer Funktion `average(values)`. Gib den Durchschnitt, die beste und die schlechteste Note aus. Bonus: Zähle die ungenügenden Noten (unter 4).

Lösungen

1 Index lesen

a) 7 · b) 1 · c) 5 · d) 5 · e) IndexError: list index out of range – der grösste Index ist 4.

2 Umfang berechnen

`def perimeter(width, height): return 2 * (width + height) print(perimeter(4, 5))` → 18

3 print oder return?

Ausgabe 8, dann TypeError: Die Funktion gibt nichts zurück, `y` ist `None`, und `None + 1` geht nicht. Korrektur: `return 2 * x` statt `print(2 * x)` → Ausgabe 9.

4 Liste verändern

["Apfel", "Birne", "Kiwi"] → ["Banane", "Apfel", "Birne", "Kiwi"] → ["Banane", "Apfel", "Kiwi"]

5 Schreibtischtest

v = 4 → n = 0 · v = 12 → n = 1 · v = 15 → n = 2 · v = 8 → n = 2 · v = 20 → n = 3. Ergebnis 3.

6 Notenrechner

`def average(values): return sum(values) / len(values) grades = [5, 4.5, 6, 3.5] print(average(grades), max(grades), min(grades))` → 4.75, 6, 3.5. Bonus: `low = 0 for g in grades: if g < 4: low = low + 1` → 1.

Q Quiz im Video

1 B (return) · 2 A (0) · 3 C (hängt hinten an) · 4 A (3)