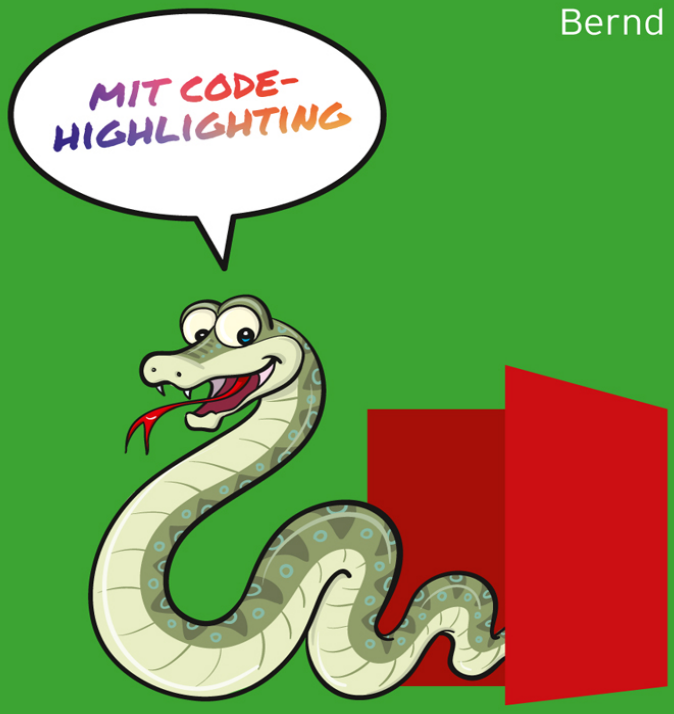
Bernd KLEIN



EINFÜHRUNG IN PYTHON 3

FÜR EIN- UND UMSTEIGER

4. Auflage

Im Internet:
Musterlösungen zu den Übungen

HANSER

HANSER

Bernd Klein

Einführung in Python 3

Für Ein- und Umsteiger

4., vollständig überarbeitete Auflage

Der Autor:

Bernd Klein, bernd@python-kurs.eu

Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Lektorat: Brigitte Bauer-Schiewek
Copy editing: Jürgen Dubau, Freiburg/Elbe

Layout: le-tex publishing services, Leipzig
Umschlagdesign: Marc Müller-Bremer, www.rebranding.de, München
Umschlagrealisation: Max Kostopoulos
Titelmotiv: © istockphoto.com/zaricm
Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702

Print-ISBN: 978-3-446-46379-0
E-Book-ISBN: 978-3-446-46556-5
E-Pub-ISBN: 978-3-446-46467-4

Inhalt

Titelei

Impressum

Inhalt

Vorwort

Danksagung

Teil I: Einleitung

1 Einleitung

1.1 Einfach und schnell zu lernen

1.2 Geschichte von Python

1.3 Zen von Python

1.4 Zielgruppe des Buches

1.5 Aufbau des Buches

1.6 Programmieren lernen „interaktiv”

1.7 Download der Beispiele und Hilfe

1.8 Anregungen und Kritik

Teil II: Grundlagen

2 Kommandos und Programme

2.1 Erste Schritte mit Python

2.1.1 Linux

2.1.2 Windows

2.1.3 macOS

2.2 Herkunft und Bedeutung des Begriffes interaktive Shell

2.2.1 Erste Schritte in der interaktiven Shell

2.3 Verlassen der Python-Shell

2.4 Benutzung von Variablen

2.5 Mehrzeilige Anweisungen in der interaktiven Shell

2.6 Programme schreiben oder schnell mal der Welt “Hallo” sagen

3 Bytecode und Maschinencode

3.1 Einführung

3.2 Unterschied zwischen Programmier- und Skriptsprachen

3.3 Interpreter- oder Compilersprache

4 Datentypen und Variablen

4.1 Einführung

4.2 Variablennamen

4.2.1 Gültige Variablennamen

4.2.2 Konventionen für Variablennamen

4.3 Datentypen

4.3.1 Ganze Zahlen

4.3.2 Fließkommazahlen

4.3.3 Zeichenketten

4.3.4 Boolesche Werte

4.3.5 Komplexe Zahlen

4.3.6 Operatoren

4.4 Statische und dynamische Typdeklaration

4.5 Typumwandlung

4.6 Datentyp ermitteln

5 Sequentielle Datentypen

5.1 Übersicht

5.1.1 Zeichenketten oder Strings

5.1.2 Listen

5.1.3 Tupel

5.2 Indizierung von sequentiellen Datentypen

5.3 Teilbereichsoperator

5.4 Die len-Funktion

5.5 Aufgaben

6 Listen und Tupel im Detail

6.1 Virtueller Einkaufsbummel

6.2 Stapelspeicher/ Stacks

6.3 Stapelverarbeitung in Python: pop und append

6.4 extend

6.5 Module importieren

6.6 ‚+'-Operator oder append

6.7 Entfernen eines Wertes

6.8 Prüfen, ob ein Element in Liste enthalten ist

6.9 Finden der Position eines Elementes

6.10 Einfügen von Elementen

6.11 Besonderheiten bei Tupel

6.11.1 Leere Tupel

6.11.2 1-Tupel

6.11.3 Mehrfachzuweisungen, Packing und Unpacking

6.12 Die veränderliche Unveränderliche

6.13 Aufgaben

7 Verzweigungen

7.1 Anweisungsblöcke und Einrückungen

7.2 Bedingte Anweisungen in Python

7.2.1 Einfachste if-Anweisung

7.2.2 if-Anweisung mit else-Zweig

7.2.3 elif-Zweige

7.3 Vergleichsoperatoren

7.4 Zusammengesetzte Bedingungen

7.5 Wahr oder falsch: Bedingungen in Verzweigungen

7.6 Aufgaben

8 Schleifen

8.1 Übersicht

8.2 while-Schleife

8.3 break und continue

8.4 Die Alternative im Erfolgsfall: else

8.5 Iterierbare Objekte (iterables)

8.6 For-Schleife

8.7 Aufgaben

9 Dictionaries

9.1 Definition und Benutzung

9.2 Fehlerfreie Zugriffe auf Dictionaries

9.3 Einfachere Definition von Dictionaries

9.4 Zulässige Typen für Schlüssel und Werte

9.5 Verschachtelte Dictionaries

9.6 Dictionaries in Listen wandeln

9.7 Weitere Methoden auf Dictionaries

9.8 Operatoren

9.9 Die zip-Funktion

9.10 Dictionaries aus Listen erzeugen

9.11 Aufgaben

10 Mengen

10.1 Übersicht

10.2 Mengen in Python

10.2.1 Sets erzeugen

10.2.2 Mengen von unveränderlichen Elementen

10.3 Frozensets

10.4 Operationen auf „set”-Objekten

10.4.1 add(element)

10.4.2 clear()

10.4.3 copy

10.4.4 difference()

10.4.5 difference_update()

10.4.6 discard(el)

10.4.7 remove(el)

10.4.8 intersection(s)

10.4.9 union(s)

10.4.10 isdisjoint()

10.4.11 issubset()

10.4.12 issuperset()

10.4.13 pop()

10.5 Erweiterte Zuweisungsoperatoren für Mengen

11 Eingaben

11.1 Eingabe mittels input

12 Dateien lesen und schreiben

12.1 Dateien

12.2 Text aus einer Datei lesen

12.3 Schreiben in eine Datei

12.4 In einem Rutsch lesen: readlines und read

12.5 with-Anweisung

12.6 Aufgaben

13 Formatierte Ausgabe und Strings formatieren

13.1 Wege, die Ausgabe zu formatieren

13.2 print-Funktion

13.3 Notwendigkeit

13.4 Formatierte Stringliterale / f-Strings

13.5 Die String-Methode „format”

13.6 Stringmodulo-Operator oder Formatierung à la C

13.7 Benutzung von Dictionaries beim Aufruf der „format”-Methode

13.8 Benutzung von lokalen Variablen in „format”

13.9 Weitere String-Methoden zum Formatieren

14 Referenzen, flaches und tiefes Kopieren

14.1 Einführung

14.2 Swen und Sarah

14.3 Variablen sind Referenzen

14.4 Kopieren einer Liste

14.5 Flache Kopien

14.6 Kopieren mit deepcopy

14.7 Problemverhinderung

14.8 Deepcopy für Dictionaries

15 Funktionen

15.1 Allgemein

15.2 Funktionen

15.3 Docstring

15.4 Standardwerte für Funktionen

15.5 Schlüsselwortparameter

15.6 Funktionen ohne oder mit leerer return-Anweisung

15.7 Mehrere Rückgabewerte

15.8 Parameterübergabe im Detail

15.9 Effekte bei veränderlichen Objekten

15.10 Kommandozeilenparameter

15.11 Variable Anzahl von Parametern / Variadische Funktionen

15.12 * in Funktionsaufrufen

15.13 Beliebige Schlüsselwortparameter

15.14 Doppeltes Sternchen im Funktionsaufruf

15.15 Aufgaben

16 Wertebereich von Variablen

16.1 Einführung

16.2 Globale und lokale Variablen in Funktionen

16.3 nonlocal-Variablen

17 Rekursive Funktionen

17.1 Definition und Herkunft des Begriffs

17.2 Definition der Rekursion

17.3 Rekursive Funktionen in Python

17.4 Die Tücken der Rekursion

17.5 Fibonacci-Folge in Python

17.6 Aufgaben

18 Sortieren

18.1 Sortieren von Listen

18.1.1 „sort” und „sorted”

18.1.2 Umkehrung der Sortierreihenfolge

18.1.3 Eigene Sortierfunktionen

18.2 Aufgaben

19 Modularisierung

19.1 Module

19.1.1 Namensräume von Modulen

19.1.2 Namensräume umbenennen

19.1.3 Modularten

19.1.4 Suchpfad für Module

19.1.5 Inhalt eines Moduls

19.1.6 Eigene Module

19.1.7 Dokumentation für eigene Module

19.2 Pakete

19.2.1 Einfaches Paket erzeugen

19.2.2 Komplexeres Paket

19.2.3 Komplettes Paket importieren

20 Alles über Strings

20.1 ... fast alles

20.2 Aufspalten von Zeichenketten

20.2.1 split

20.2.2 rsplit

20.2.3 Folge von Trennzeichen

20.2.4 splitlines

20.2.5 partition und rpartition

20.3 Zusammenfügen von Stringlisten mit join

20.4 Suchen von Teilstrings

20.4.1 „in” oder „not in”

20.4.2 s.find(substring[, start[, end]])

20.4.3 s.rfind(substring[, start[, end]])

20.4.4 s.index(substring[, start[, end]])

20.4.5 s.rindex(substring[, start[, end]])

20.4.6 s.count(substring[, start[, end]])

20.5 Suchen und Ersetzen

20.6 Nur noch Kleinbuchstaben oder Großbuchstaben

20.7 capitalize und title

20.8 Stripping Strings

20.9 Strings ausrichten

20.10 String-Tests

20.11 Aufgaben

21 Ausnahmebehandlung

21.1 Abfangen mehrerer Ausnahmen

21.2 except mit mehrfachen Ausnahmen

21.3 Die optionale else-Klausel

21.4 Fehlerinformationen über sys.exc_info

21.5 Exceptions generieren

21.6 Finalisierungsaktion

Teil III: Objektorientierte Programmierung

22 Grundlegende Aspekte

22.1 Bibliotheksvergleich

22.2 Objekte und Instanzen einer Klasse

22.3 Kapselung von Daten und Methoden

22.4 Eine minimale Klasse in Python

22.5 Eigenschaften und Attribute

22.6 Methoden

22.7 Instanzvariablen

22.8 Die __init__-Methode

22.9 Destruktor

22.10 Datenkapselung, Datenabstraktion und Geheimnisprinzip

22.10.1 Definitionen

22.10.2 Zugriffsmethoden

22.10.3 Properties

22.10.4 Public-, Protected- und Private-Attribute

22.10.5 Weitere Möglichkeiten der Properties

22.10.6 Properties mit Dekorateuren

22.11 Stringausgaben mit str und repr

22.12 Klassenattribute

22.13 Statische Methoden

22.13.1 Einleitendes Beispiel

22.13.2 Getter und Setter für private Klassenattribute

22.14 Public-Attribute statt private Attribute

22.15 Magische Methoden und Operatorüberladung

22.15.1 Einführung

22.15.2 Übersicht magische Methoden

22.15.3 Beispielklasse: Length

23 Bruchklasse

23.1 Brüche à la 1001 Nacht

23.2 Zurück in die Gegenwart

23.3 Rechenregeln

23.3.1 Multiplikation von Brüchen

23.3.2 Division von Brüchen

23.3.3 Addition von Brüchen

23.3.4 Subtraktion von Brüchen

23.3.5 Vergleichsoperatoren

23.4 Integer plus Bruch

23.4.1 Die Bruchklasse im Überblick

23.5 Fraction-Klasse

24 Aufrufbare Objekte

24.1 Einführung

24.1.1 Die callable-Funktion

24.1.2 Klassen statt Funktionen

25 Vererbung

25.1 Oberbegriffe und Oberklassen

25.2 Ein einfaches Beispiel

25.3 Überladen, Überschreiben und Polymorphie

25.4 Vererbung in Python

25.5 Klassenmethoden

25.6 Standardklassen als Basisklassen

26 Mehrfachvererbung

26.1 Einführung

26.2 Beispiel: KalenderUndUhr

26.3 Diamand-Problem oder „deadly diamond of death”

26.4 super und MRO

26.4.1 Ein einfaches Beispiel

26.4.2 Ein umfangreicheres Beispiel

27 Slots

27.1 Erzeugung von dynamischen Attributen verhindern

28 Dynamische Erzeugung von Klassen

28.1 Beziehung zwischen „class” und „type”

29 Metaklassen

29.1 Motivation

29.2 Definition

29.3 Definition von Metaklassen in Python

29.4 Singletons mit Metaklassen erstellen

29.5 Beispiel: Methodenaufrufe zählen

29.5.1 Einführung

29.5.2 Vorbereitungen

29.5.3 Ein Dekorateur, um Funktionsaufrufe zu zählen

29.5.4 Die Metaklasse „Aufrufzähler”

30 Abstrakte Klassen

31 Aufgaben zur Objektorientierung

Teil IV: Funktionale Programmierung

32 Begriffsbestimmung

33 lambda, map, filter und reduce

33.1 lambda

33.2 map

33.3 Filtern von sequentiellen Datentypen mittels „filter”

33.4 reduce

33.5 Aufgaben

34 Listen-Abstraktion/List Comprehension

- 34.1 Die Alternative zu Lambda und Co.
- 34.2 Syntax
- 34.3 Weitere Beispiele
- 34.4 Die zugrunde liegende Idee
- 34.5 Anspruchsvolleres Beispiel
- 34.6 Mengen-Abstraktion
- 34.7 Rekursive Primzahlberechnung
- 34.8 Generatoren-Abstraktion
- 34.9 Aufgaben

35 Generatoren und Iteratoren

- 35.1 Einführung
- 35.2 Iteration in for-Schleifen
- 35.3 Generatoren
- 35.4 Endlos-Generatoren zähmen mit firstn und islice
- 35.5 Sinnvollere Beispiele
- 35.6 Beispiele aus der Kombinatorik

35.6.1 Permutationen

35.6.2 Variationen und Kombinationen

35.7 Generator-Ausdrücke

35.8 return-Anweisungen in Generatoren

35.9 send-Methode

35.10 Die close-Methode

35.11 Die throw-Methode

35.12 Dekoration von Generatoren

35.13 yield from

35.14 Aufgaben

36 Dekorateure

36.1 Einführung Dekorateure

36.1.1 Verschachtelte Funktionen

36.1.2 Funktionen als Parameter

36.1.3 Funktionen als Rückgabewert

36.1.4 Fabrikfunktionen

36.2 Ein einfacher Dekorateur

36.3 @-Syntax für Dekorateure

36.4 Anwendungsfälle für Dekorateure

36.4.1 Überprüfung von Argumenten durch Dekorateure

36.4.2 Funktionsaufrufe mit einem Dekorateur zählen

36.5 Dekorateure mit Parametern

36.6 Benutzung von Wraps aus functools

36.7 Eine Klasse als Dekorateur benutzen

36.8 Memoisation

36.8.1 Bedeutung und Herkunft des Begriffs

36.8.2 Memoisation mit Dekorateurfunktionen

36.8.3 Memoisation mit einer Klasse

36.8.4 Memoisation mit functools.lru_cache

Teil V: Weiterführende Themen

37 Tests und Fehler

37.1 Einführung

37.2 Modultests

37.3 Modultests unter Benutzung von __name__

37.4 doctest-Modul

37.5 Testgetriebene Entwicklung oder „Im Anfang war der Test”

37.6 unittest

37.7 Methoden der Klasse TestCase

37.8 Aufgaben

38 Daten konservieren

38.1 Persistente Speicherung

38.2 Pickle-Modul

38.2.1 Daten „einpökeln” mit pickle.dump

38.2.2 pickle.load

38.3 Ein persistentes Dictionary mit shelve

39 Reguläre Ausdrücke

39.1 Ursprünge und Verbreitung

39.2 Stringvergleiche

39.3 Überlappungen und Teilstrings

39.4 Das re-Modul

39.5 Matching-Problem

39.6 Syntax der regulären Ausdrücke

39.6.1 Beliebiges Zeichen

39.7 Zeichenauswahl

39.8 Endliche Automaten

39.9 Anfang und Ende eines Strings

39.10 Vordefinierte Zeichenklassen

39.11 Optionale Teile

39.12 Quantoren

39.13 Gruppierungen und Rückwärtsreferenzen

39.13.1 Match-Objekte

39.14 Iteration über Matches mit finditer

39.15 Umfangreiche Übung

39.16 Alles finden mit findall

39.17 Alternativen

39.18 Compilierung von regulären Ausdrücken

39.19 Aufspalten eines Strings mit oder ohne regulären Ausdruck

39.19.1 split-Methode der String-Klasse

39.19.2 split-Methode des re-Moduls

39.19.3 Wörter filtern

39.20 Suchen und Ersetzen mit sub

39.21 Aufgaben

40 Typanmerkungen

40.1 Einführung

40.2 Einfaches Beispiel

40.3 Variablenanmerkungen

40.4 Listenbeispiele

40.5 Listen mit homogenem Typ

40.5.1 Version 3.6 bis 3.8

40.5.2 Python 3.9 und später

41 Systemprogrammierung

41.1 Einleitung

41.2 Häufig falsch verstanden: Shell

41.3 os-Modul

41.3.1 Vorbemerkungen

41.3.2 Umgebungsvariablen

41.3.3 Dateiverarbeitung auf niedrigerer Ebene

41.3.4 Weitere Funktionen im Überblick

41.3.5 os.path – Arbeiten mit Pfaden

41.4 shutil-Modul

41.5 glob-Modul

42 Forks

42.1 Fork

42.2 Fork in Python

Teil VI: Lösungen zu den Aufgaben

43 Lösungen zu den Aufgaben

43.1 Lösungen zu Kapitel 5 (Sequentielle Datentypen)

43.2 Lösungen zu Kapitel 6 (Listen und Tupel im Detail)

43.3 Lösungen zu Kapitel 7 (Verzweigungen)

43.4 Lösungen zu Kapitel 8 (Schleifen)