

Michael Weigend

Objektorientierte Programmierung mit Python

3. aktualisierte und
erweiterte Auflage



Inhaltsverzeichnis

	Einleitung	21
	Warum Python?	21
	An wen wendet sich dieses Buch?	21
	Inhalt und Aufbau	22
	Hinweise zur Typographie	23
	Programmbeispiele	23
I	Grundlagen	25
I.1	Was ist Programmieren?	25
I.2	Hardware und Software	26
I.3	Programm als Algorithmus	27
I.4	Syntax und Semantik	28
I.5	Interpreter und Compiler	28
I.6	Programmierparadigmen	30
I.7	Objektorientierte Programmierung	31
I.7.1	Strukturelle Zerlegung	31
I.7.2	Die Welt als System von Objekten	32
I.7.3	Objekte besitzen Attribute und beherrschen Methoden	33
I.7.4	Objekte sind Instanzen von Klassen	34
I.8	Geschichte der objektorientierten Programmierung	34
I.9	Aufgaben	35
I.10	Lösungen	35
2	Python im interaktiven Modus	37
2.1	Python installieren	37
2.2	Python im interaktiven Modus	38
2.2.1	Start des Python-Interpreters in einem Konsole-Fenster ...	39
2.2.2	Die Python-Shell von IDLE	39
2.2.3	Die ersten Python-Befehle ausprobieren	39
2.2.4	Hotkeys	40
2.3	Objekte	41

2.4	Namen	43
2.4.1	Syntax-Regeln für Bezeichner	43
2.4.2	Schlüsselwörter	44
2.5	Anweisungen	44
2.5.1	Ausdrucksanweisungen	45
2.5.2	Import-Anweisungen	48
2.5.3	Zuweisungen	49
2.5.4	Erweiterte Zuweisungen	53
2.5.5	Ausgabe von Werten – die print-Anweisung	53
2.6	Aufgaben	54
2.7	Lösungen	56
3	Python-Skripte	59
3.1	Skripte editieren und ausführen mit IDLE	59
3.2	Ausführen eines Python-Skripts	60
3.3	Kommentare	63
3.4	Die Zeilenstruktur von Python-Programmen	63
3.5	Das EVA-Prinzip	67
3.6	Phasen der Programmentwicklung	69
3.7	Guter Programmierstil	70
3.8	Die Kunst des Fehlerfindens	72
3.9	Aufgaben	74
3.10	Lösungen	75
4	Datentypen	77
4.1	Daten als Objekte	77
4.2	Fundamentale Datentypen im Überblick	79
4.3	Datentypen und Klassen	80
4.4	NoneType	81
4.5	Wahrheitswerte – der Datentyp bool	81
4.6	Ganze Zahlen	82
4.6.1	int und long	83
4.6.2	Dezimalzahlen, Oktalzahlen und Hexadezimalzahlen	83
4.7	Gleitkommazahlen	84
4.8	Komplexe Zahlen	86
4.9	Arithmetische Operatoren für Zahlen	87

4.10	Sequenzen	93
4.10.1	Zeichenketten (Strings)	94
4.10.2	Unicode-Strings	95
4.10.3	Raw-Strings	96
4.10.4	Tupel	97
4.10.5	Liste	98
4.10.6	Einige Grundoperationen für Sequenzen	99
4.10.7	Veränderbare und unveränderbare Sequenzen	101
4.11	Mengen	102
4.12	Dictionaries	103
4.13	Typumwandlungen	104
4.13.1	int()	105
4.13.2	float()	106
4.13.3	complex()	106
4.13.4	bool()	106
4.13.5	str()	107
4.13.6	unicode()	107
4.13.7	dict(), list() und tuple()	108
4.14	Aufgaben	109
4.15	Lösungen	112
5	Kontrollstrukturen	117
5.1	Einfache Bedingungen	117
5.1.1	Vergleiche	117
5.1.2	Zugehörigkeit zu einer Menge (in, not in)	121
5.1.3	Beliebige Ausdrücke als Bedingungen	121
5.2	Zusammengesetzte Bedingungen – logische Operatoren	122
5.2.1	Negation (not)	122
5.2.2	Konjunktion (and)	123
5.2.3	Disjunktion (or)	124
5.2.4	Formalisierung von Bedingungen	125
5.2.5	Hinweis zum Programmierstil	125
5.3	Programmverzweigungen (bedingte Anweisungen)	126
5.3.1	Einseitige Verzweigung (if)	126
5.3.2	Zweiseitige Verzweigung (if-else)	127
5.3.3	Mehrfache Fallunterscheidung (elif)	128

5.4	Bedingte Wiederholung (while)	129
5.4.1	Endlosschleifen	130
5.5	Iteration (for)	131
5.5.1	Zählschleifen – Verwendung von range()	133
5.5.2	Verschachtelte Iterationen	134
5.5.3	Iterative Berechnung rekursiver Folgen	135
5.6	Abbruch einer Schleife mit break	136
5.6.1	Abbruch eines Schleifendurchlaufs mit continue	137
5.7	Abfangen von Ausnahmen mit try	137
5.7.1	try...except	138
5.7.2	try...finally	140
5.8	Aufgaben	141
5.9	Lösungen	146
6	Funktionen	151
6.1	Aufruf von Funktionen	151
6.2	Definition von Funktionen	153
6.3	Schrittweise Verfeinerung	154
6.4	Ausführung von Funktionen	158
6.4.1	Globale und lokale Namen	158
6.4.2	Seiteneffekte – die global-Anweisung	161
6.4.3	Parameterübergabe	162
6.5	Voreingestellte Parameterwerte	164
6.5.1	Schlüsselwort-Argumente	166
6.6	Funktionen mit beliebiger Anzahl von Parametern	168
6.7	Lokale Funktionen	169
6.8	Rekursive Funktionen	170
6.8.1	Execution Frames	172
6.8.2	Rekursionstiefe	173
6.9	Funktionen als Objekte	175
6.10	Lambda-Formen	176
6.11	Hinweise zum Programmierstil	176
6.11.1	Allgemeines	176
6.11.2	Funktionsnamen	177
6.11.3	Kommentierte Parameter	177
6.11.4	Docstrings	177
6.12	Aufgaben	178
6.13	Lösungen	181

7	Sequenzen, Mengen und Generatoren	185
7.1	Gemeinsame Operationen für Sequenzen	185
7.1.1	Zugriff auf Elemente einer Sequenz	186
7.1.2	Slicing von Sequenzen	187
7.1.3	Anwendung von Slicing bei rekursiven Algorithmen	188
7.1.4	Rekursive Suche in einer Sequenz	188
7.2	Tupel	190
7.3	Listen	191
7.3.1	Eine Liste erzeugen	192
7.3.2	Eine Liste verändern	194
7.3.3	Flache und tiefe Kopien	196
7.3.4	Listen sortieren	198
7.3.5	Binäre Suche in einer sortierten Liste	199
7.3.6	Zwei Sortierverfahren im Vergleich	200
7.3.7	Modellieren mit Listen – Beispiel: die Charts	204
7.4	Generatoren	208
7.4.1	Generatorausdrücke	209
7.4.2	Generatorfunktionen	209
7.4.3	Iteratoren	211
7.4.4	Verwendung von Generatoren	211
7.5	Mengen	212
7.5.1	Operationen für Mengen	213
7.5.2	Modellieren mit Mengen – Beispiel: Graphen	214
7.6	Aufgaben	217
7.7	Lösungen	219
8	Dictionaries	223
8.1	Operationen für Dictionaries	223
8.2	Wie erstellt man ein Dictionary?	224
8.2.1	Definition mit einem Dictionary-Display	224
8.2.2	Schrittweiser Aufbau eines Dictionarys	226
8.2.3	Ein Dictionary aus anderen Dictionaries zusammensetzen – update()	226
8.3	Zugriff auf Daten in einem Dictionary	227
8.3.1	Vergebliche Zugriffsversuche	227
8.4	Praxisbeispiel: Vokabeltrainer	228
8.5	Typische Fehler	230
8.6	Aufgaben	230
8.7	Lösungen	233

9	Ein- und Ausgabe	237
9.1	Files	237
9.1.1	Die Rolle der Files bei E/A-Operationen	237
9.1.2	Was ist ein File?	238
9.1.3	Ein File-Objekt erzeugen	239
9.1.4	Speichern einer Zeichenkette	239
9.1.5	Laden einer Zeichenkette aus einer Datei	241
9.1.6	Absolute und relative Pfade	241
9.1.7	Zwischenspeichern ohne zu schließen	244
9.1.8	Zugriff auf Files (lesen und schreiben)	244
9.1.9	Speichern beliebiger Daten auf Files	246
9.2	Objekte speichern mit pickle	247
9.2.1	Funktionen zum Speichern und Laden	247
9.2.2	cpickle	249
9.3	Eingabe über die Tastatur	250
9.4	Die Pseudofiles sys.stdin und sys.stdout	251
9.5	Formatierte Ausgabe von Werten mit dem Formatierungsoperator %	251
9.5.1	Anwendung: Ausgabe von Tabellen	253
9.6	Kommandozeilen-Argumente (Optionen)	253
9.7	Aufgaben	256
9.8	Lösungen	259
10	Definition eigener Klassen	263
10.1	Klassen und Objekte	263
10.2	Definition von Klassen	265
10.3	Objekte (Instanzen)	267
10.4	Zugriff auf Attribute – Sichtbarkeit	270
10.4.1	Öffentliche Attribute	270
10.4.2	Private Attribute	271
10.4.3	Properties	274
10.4.4	Dynamische Erzeugung von Attributen	275
10.5	Methoden	275
10.5.1	Polymorphismus – Überladen von Operatoren	276
10.6	Statische Methoden	280
10.7	Abstraktion, Verkapselung und Geheimnisprinzip	281

10.8	Vererbung	282
10.8.1	Spezialisierungen	282
10.8.2	Beispiel: Die Klasse Konto – eine Spezialisierung der Klasse Geld	283
10.8.3	Standardklassen als Basisklassen	286
10.9	Hinweise zum Programmierstil	288
10.9.1	Bezeichner	288
10.9.2	Sichtbarkeit	288
10.9.3	Dokumentation von Klassen	290
10.10	Typische Fehler	290
10.11	Aufgaben	292
10.12	Lösungen	295
11	Klassenbibliotheken in Modulen speichern	299
11.1	Testen einer Klasse in einem lauffähigen Stand-alone-Skript	299
11.2	Module speichern und importieren	301
11.3	Den Zugang zu einem Modul sicherstellen	302
11.4	Kompilieren von Modulen	303
11.5	Programmierstil: Verwendung und Dokumentation von Modulen ..	304
12	Objektorientiertes Modellieren	305
12.1	Phasen einer objektorientierten Software-Entwicklung	305
12.2	Fallstudie: Modell eines Wörterbuchs	306
12.2.1	OOA: Entwicklung einer Klassenstruktur	306
12.2.2	OOD: Entwurf einer Klassenstruktur für eine Implementierung in Python	310
12.2.3	OOP: Implementierung der Klassenstruktur	312
12.3	Assoziationen zwischen Klassen	316
12.3.1	Reflexive Assoziationen	316
12.3.2	Aggregation	318
12.4	Beispiel: Management eines Musicals	318
12.4.1	OOA	318
12.4.2	OOD	320
12.4.3	OOP	321
12.5	Aufgaben	330
12.6	Lösungen	331

13	Verarbeitung von Zeichenketten	335
13.1	Standardmethoden zur Verarbeitung von Zeichenketten	335
13.1.1	Formatieren	336
13.1.2	Schreibweise	336
13.1.3	Tests	337
13.1.4	Entfernen und Aufspalten	337
13.1.5	Suchen und Ersetzen	338
13.1.6	Codierung und Decodierung	339
13.2	Automatische Textproduktion	344
13.2.1	Texte mit variablen Teilen – der Formatierungsoperator %	344
13.2.2	Textuelle Repräsentation eines Objektes	347
13.3	Analyse von Texten	348
13.3.1	Chat Bots	348
13.3.2	Textanalyse mit einfachen Vorkommenstests	350
13.4	Reguläre Ausdrücke	351
13.4.1	Aufbau eines regulären Ausdrucks	352
13.4.2	Objekte für reguläre Ausdrücke (RE-Objekte)	355
13.4.3	Textpassagen extrahieren mit findall()	357
13.4.4	Zeichenketten zerlegen mit split()	359
13.4.5	Teilstrings ersetzen mit sub()	360
13.4.6	Match-Objekte	360
13.5	Den Computer zum Sprechen bringen – Sprachsynthese	363
13.5.1	Links zum Thema Sprachsynthese	365
13.6	Aufgaben	365
13.7	Lösungen	368
14	Systemfunktionen	377
14.1	Das Modul sys – die Schnittstelle zum Laufzeitsystem	377
14.1.1	Informationen über die aktuelle Systemumgebung	378
14.1.2	Standardeingabe und -ausgabe	379
14.1.3	Die Objektverwaltung beobachten mit getrefcount()	380
14.1.4	Ausführung eines Skripts beenden	381
14.2	Das Modul os – die Schnittstelle zum Betriebssystem	381
14.2.1	Dateien und Verzeichnisse suchen	382
14.2.2	Zugriffsrechte abfragen und ändern (Windows und Unix)	383
14.2.3	Dateien und Verzeichnisse anlegen und modifizieren	386
14.2.4	Merkmale von Dateien und Verzeichnissen abfragen	387

14.2.5	Umgebungsvariablen	388
14.2.6	Systematisches Durchlaufen eines Verzeichnisbaumes	388
14.3	Datum und Zeit	391
14.3.1	Funktionen des Moduls time	392
14.3.2	Sekundenformat	392
14.3.3	Zeittupel	393
14.3.4	Zeitstrings	394
14.3.5	Einen Prozess unterbrechen mit sleep()	395
14.4	Aufgaben	395
14.5	Lösungen	396
15	Gestaltung von grafischen Benutzungsoberflächen	399
15.1	Ein einführendes Beispiel	400
15.2	Einfache Widgets	403
15.3	Die Master-Slave-Hierarchie	404
15.4	Optionen der Widgets	405
15.4.1	Optionen bei der Instanzierung setzen	405
15.4.2	Widget-Optionen nachträglich konfigurieren	406
15.4.3	Fonts	407
15.4.4	Farben	408
15.4.5	Rahmen	408
15.4.6	Die Größe eines Widgets	409
15.4.7	Leerraum um Text	411
15.5	Gemeinsame Methoden der Widgets	411
15.6	Die Klasse Tk	412
15.7	Die Klasse Button	413
15.8	Die Klasse Label	413
15.8.1	Dynamische Konfiguration der Beschriftung	413
15.8.2	Verwendung von Kontrollvariablen	415
15.9	Die Klasse Entry	416
15.10	Die Klasse Radiobutton	418
15.11	Die Klasse Checkbutton	420
15.12	Die Klasse Scale	422
15.13	Die Klasse Frame	424
15.14	Aufgaben	424
15.15	Lösungen	426

16	Layout	431
16.1	Der Packer	431
16.2	Layout-Fehler	433
16.3	Raster-Layout	434
16.4	Vorgehensweise bei der GUI-Entwicklung	438
16.4.1	Die Benutzungsoberfläche gestalten	440
16.4.2	Funktionalität hinzufügen	443
16.5	Aufgaben	444
16.6	Lösungen	445
17	Grafik	453
17.1	Die Tkinter-Klasse Canvas	453
17.1.1	Generierung grafischer Elemente – ID, Positionierung und Display-Liste	454
17.1.2	Grafische Elemente gestalten	456
17.1.3	Visualisieren mit Kreisdiagrammen	458
17.2	Die Klasse PhotoImage	461
17.2.1	Eine Pixelgrafik erzeugen	462
17.2.2	Fotos analysieren und verändern	464
17.3	Bilder in eine Benutzungsoberfläche einbinden	466
17.3.1	Icons auf Schaltflächen	467
17.3.2	Hintergrundbilder	468
17.4	Aufgaben	470
17.5	Lösungen	471
18	Event-Verarbeitung	475
18.1	Einführendes Beispiel	476
18.2	Event-Sequenzen	477
18.2.1	Event-Typen	478
18.2.2	Qualifizierer für Maus- und Tastatur-Events	478
18.2.3	Modifizierer	479
18.3	Beispiel: Tastaturereignisse verarbeiten	480
18.4	Programmierung eines Eventhandlers	482
18.4.1	Beispiel für eine Event-Auswertung	482
18.5	Bindemethoden	483
18.6	Aufgaben	484
18.7	Lösungen	486

19	Komplexe Benutzungsoberflächen	491
19.1	Text-Widgets	491
19.1.1	Methoden der Text-Widgets	492
19.2	Rollbalken (Scrollbars)	494
19.3	Menüs	495
19.3.1	Die Klasse Menu	496
19.3.2	Methoden der Klasse Menu	496
19.4	Texteditor mit Menüleiste und Pulldown-Menü	498
19.5	Dialogboxen	499
19.6	Aufgaben	503
19.7	Lösungen	503
20	Threads	507
20.1	Funktionen in einem Thread ausführen	508
20.2	Thread-Objekte erzeugen – die Klasse Thread	510
20.3	Aufgaben	513
20.4	Lösungen	514
21	Fehler finden und vermeiden	519
21.1	Testen von Bedingungen	519
21.1.1	Ausnahmen (Exceptions)	519
21.1.2	Testen von Vor- und Nachbedingungen mit assert()	521
21.1.3	Testen im Debugging-Modus	523
21.1.4	Ausnahmen gezielt auslösen	523
21.2	Selbstdokumentation	524
21.3	Debugging	525
21.4	Aufgabe	527
21.5	Lösung	528
22	CGI-Programmierung	529
22.1	Wie funktionieren CGI-Skripte?	529
22.2	Aufbau eines einfachen CGI-Skripts	531
22.3	CGI-Skripte ausführen	532
22.4	Kommunikation über interaktive Webseiten	534
22.4.1	Aufbau eines HTML-Formulars	535
22.5	Verarbeitung von Eingabedaten in einem CGI-Skript	538
22.6	CGI-Skripte debuggen	540
22.7	Objektorientierte CGI-Skripte – Beispiel: ein Chat-Room	541

22.8	CGI-Skripte mit Cookies	546
22.9	Aufgaben	549
22.10	Lösungen	551
23	Internet-Programmierung	555
23.1	Was ist ein Protokoll?	555
23.2	Übertragung von Dateien mit FTP	556
23.2.1	Das Modul ftplib	557
23.2.2	Navigieren und Downloaden	558
23.2.3	Ein Suchroboter für FTP-Server	559
23.3	Zugriff auf Webseiten mit HTTP	563
23.3.1	Automatische Auswertung von Webseiten	565
23.4	E-Mails senden mit SMTP	566
23.5	Aufgaben	569
23.6	Lösungen	571
24	Datenbanken	577
24.1	Was ist ein Datenbanksystem?	577
24.2	Entity-Relationship-Diagramme (ER-Diagramme)	578
24.3	Relationale Datenbanken	579
24.4	Darstellung von Relationen als Listen oder Dictionaries	580
24.5	Das Modul anydbm – Zugang zu DBM-Datenbanken	581
24.5.1	Ein Datenbank-Objekt erzeugen	581
24.5.2	Zugriff auf Datenbank-Objekte	581
24.5.3	Wie speichert man Tupel?	582
24.6	Online-Redaktionssystem mit Datenbankanbindung	583
24.6.1	Objektorientierte Analyse (OOA)	585
24.6.2	Objektorientierter Entwurf (OOD)	585
24.6.3	Exkurs: Authentifizieren mit MD5-Fingerprints	586
24.6.4	Implementierung mit Python (OOP)	587
24.7	Aufgaben	594
24.8	Lösungen	595
25	Fortgeschrittene Programmiertechniken – Testen und Tuning	597
25.1	Automatisiertes Testen	597
25.2	Testen mit Docstrings – das Modul doctest	598
25.3	Praxisbeispiel: Suche nach dem Wort des Jahres	600

25.4	Klassen testen mit doctest	607
25.4.1	Wie testet man eine Klasse?	607
25.4.2	Normalisierte Whitespaces – doctest-Direktiven	608
25.4.3	Ellipsen verwenden	608
25.4.4	Dictionaries testen	609
25.5	Gestaltung von Testreihen mit unittest	609
25.5.1	Einführendes Beispiel mit einem Testfall	609
25.5.2	Klassen des Moduls unittest	611
25.5.3	Weiterführendes Beispiel	613
25.6	Tuning	616
25.6.1	Performanzanalyse mit dem Profiler	617
25.6.2	Praxisbeispiel: Auswertung astronomischer Fotografien ...	619
25.6.3	Performanzanalyse und Tuning	625
25.7	Aufgaben	626
25.8	Lösungen	628
26	XML	635
26.1	Was ist XML?	635
26.2	XML-Dokumente	636
26.3	Ein XML-Dokument als Baum	638
26.4	DOM	639
26.5	Das Modul xml.dom.minidom	642
26.5.1	XML-Dokumente und DOM-Objekte	642
26.5.2	Die Basisklasse Node	644
26.5.3	Die Klassen Document, Element und Text	646
26.6	Attribute von XML-Elementen	648
26.7	Anwendungsbeispiel 1: Eine XML-basierte Klasse	648
26.8	Anwendungsbeispiel 2: Datenkommunikation mit XML	652
26.8.1	Überblick	652
26.8.2	Das Client-Programm	653
26.8.3	Das Server-Programm	656
26.9	Aufgaben	661
26.10	Lösungen	662

A	Anhang	665
A.1	Zeichencodierung	665
A.1.1	Codierung von Sonderzeichen in HTML	665
A.1.2	Oktettcodierung ISO-8859-1 (erweiterter ASCII-Code, Unicode 0-255)	665
A.2	Quellen im WWW	670
A.3	Standardfunktionen	670
A.4	Mathematische Funktionen	673
A.4.1	Das Modul math	673
A.4.2	Das Modul random	674
A.5	EBNF-Grammatik	674
A.6	Einige deutsche Webhosting-Dienste, die CGI-Skripting mit Python unterstützen	678
B	Glossar	679
C	Inhalt der CD	689
	Stichwortverzeichnis	691