

Inhaltsverzeichnis

	Einleitung	19
	Warum Python?	19
	Python 3	19
	An wen wendet sich dieses Buch?	19
	Inhalt und Aufbau	20
	Hinweise zur Typographie	21
	Programmbeispiele	21
I	Grundlagen	23
I.1	Was ist Programmieren?	23
I.2	Hardware und Software	24
I.3	Programm als Algorithmus	25
I.4	Syntax und Semantik	26
I.5	Interpreter und Compiler	26
I.6	Programmierparadigmen	28
I.7	Objektorientierte Programmierung	29
	I.7.1 Strukturelle Zerlegung	29
	I.7.2 Die Welt als System von Objekten	30
	I.7.3 Objekte besitzen Attribute und beherrschen Methoden	31
	I.7.4 Objekte sind Instanzen von Klassen	32
I.8	Geschichte der objektorientierten Programmierung	32
I.9	Aufgaben	33
I.10	Lösungen	34
2	Python im interaktiven Modus	35
2.1	Python installieren	35
2.2	Python im interaktiven Modus	38
	2.2.1 Start des Python-Interpreters in einem Konsole-Fenster ...	38
	2.2.2 Die Python-Shell von IDLE	38
	2.2.3 Die ersten Python-Befehle ausprobieren	39
	2.2.4 Hotkeys	39

2.3	Objekte	40
2.4	Namen	42
2.4.1	Syntax-Regeln für Bezeichner	42
2.4.2	Schlüsselwörter	43
2.5	Anweisungen	44
2.5.1	Ausdruckanweisungen	45
2.5.2	Import-Anweisungen	49
2.5.3	Zuweisungen	50
2.5.4	Erweiterte Zuweisungen	53
2.6	Aufgaben	54
2.7	Lösungen	55
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	68
3.7	Guter Programmierstil	69
3.8	Die Kunst des Fehlerfindens	72
3.9	Aufgaben	74
3.10	Lösungen	75
4	Standard-Datentypen	77
4.1	Daten als Objekte	77
4.2	Fundamentale Datentypen im Überblick	79
4.3	Typen und Klassen	80
4.4	NoneType	81
4.5	Wahrheitswerte – der Datentyp bool	81
4.6	Ganze Zahlen	82
4.7	Gleitkommazahlen	84
4.8	Komplexe Zahlen	85
4.9	Arithmetische Operatoren für Zahlen	86
4.10	Sequenzen	91
4.10.1	Zeichenketten (Strings)	92
4.10.2	Bytestrings	94
4.10.3	Tupel	95

4.10.4	Liste	96
4.10.5	Bytearray	97
4.10.6	Einige Grundoperationen für Sequenzen	97
4.10.7	Veränderbare und unveränderbare Sequenzen	100
4.11	Mengen	101
4.12	Dictionaries	102
4.13	Typumwandlungen	102
4.13.1	int()	103
4.13.2	float()	104
4.13.3	complex()	105
4.13.4	bool()	105
4.13.5	str()	105
4.13.6	dict(), list() und tuple()	106
4.14	Aufgaben	106
4.15	Lösungen	109
5	Kontrollstrukturen	113
5.1	Einfache Bedingungen	113
5.1.1	Vergleiche	113
5.1.2	Zugehörigkeit zu einer Menge (in, not in)	117
5.1.3	Beliebige Ausdrücke als Bedingungen	117
5.2	Zusammengesetzte Bedingungen – logische Operatoren	118
5.2.1	Negation (not)	118
5.2.2	Konjunktion (and)	119
5.2.3	Disjunktion (or)	120
5.2.4	Formalisierung von Bedingungen	121
5.2.5	Hinweis zum Programmierstil	122
5.3	Programmverzweigungen (bedingte Anweisungen)	122
5.3.1	Einseitige Verzweigung (if)	123
5.3.2	Zweiseitige Verzweigung (if-else)	123
5.3.3	Mehrfache Fallunterscheidung (elif)	124
5.3.4	Bedingte Ausdrücke	126
5.4	Bedingte Wiederholung (while)	126
5.4.1	Endlosschleifen	127
5.5	Iteration über eine Kollektion (for)	129
5.5.1	Zählschleifen – Verwendung von range()	130
5.5.2	Verschachtelte Iterationen	131
5.5.3	Iterative Berechnung rekursiver Folgen	133

5.6	Abbruch einer Schleife mit break	133
5.6.1	Abbruch eines Schleifendurchlaufs mit continue	134
5.7	Abfangen von Ausnahmen mit try	135
5.7.1	try...except	136
5.8	Aufgaben	138
5.9	Lösungen	142
6	Funktionen	147
6.1	Aufruf von Funktionen	147
6.2	Definition von Funktionen	150
6.3	Schrittweise Verfeinerung	152
6.4	Ausführung von Funktionen	155
6.4.1	Globale und lokale Namen	155
6.4.2	Seiteneffekte – die global-Anweisung	159
6.4.3	Parameterübergabe	160
6.5	Voreingestellte Parameterwerte	161
6.5.1	Schlüsselwort-Argumente	164
6.6	Funktionen mit beliebiger Anzahl von Parametern	165
6.7	Lokale Funktionen	166
6.8	Rekursive Funktionen	168
6.8.1	Execution Frames	170
6.8.2	Rekursionstiefe	171
6.9	Funktionen als Objekte	173
6.9.1	Lambda-Formen	174
6.10	Hinweise zum Programmierstil	174
6.10.1	Allgemeines	174
6.10.2	Funktionsnamen	175
6.10.3	Kommentierte Parameter	175
6.10.4	Docstrings	175
6.11	Aufgaben	176
6.12	Lösungen	179
7	Sequenzen, Mengen und Generatoren	183
7.1	Gemeinsame Operationen für Sequenzen	183
7.1.1	Zugriff auf Elemente einer Sequenz	184
7.1.2	Slicing von Sequenzen	185
7.1.3	Anwendung von Slicing bei rekursiven Algorithmen	186
7.1.4	Rekursive Suche in einer Sequenz	186

7.2	Tupel	188
7.3	Listen	189
7.3.1	Eine Liste erzeugen	190
7.3.2	Eine Liste verändern	192
7.3.3	Flache und tiefe Kopien	194
7.3.4	Listen sortieren	195
7.3.5	Binäre Suche in einer sortierten Liste	197
7.3.6	Zwei Sortiervverfahren im Vergleich	198
7.3.7	Modellieren mit Listen – Beispiel: die Charts	202
7.4	Generatoren	206
7.4.1	Generatorausdrücke	207
7.4.2	Generatorfunktionen	207
7.4.3	Iteratoren	209
7.4.4	Verwendung von Generatoren	210
7.5	Mengen	211
7.5.1	Operationen für Mengen	212
7.5.2	Modellieren mit Mengen – Beispiel: Graphen	213
7.6	Aufgaben	216
7.7	Lösungen	218
8	Dictionaries	221
8.1	Operationen für Dictionaries	221
8.2	Wie erstellt man ein Dictionary?	222
8.2.1	Definition mit einem Dictionary-Display	222
8.2.2	Schrittweiser Aufbau eines Dictionaries	224
8.2.3	Ein Dictionary aus anderen Dictionaries zusammensetzen – update()	225
8.3	Zugriff auf Daten in einem Dictionary	225
8.3.1	Vergebliche Zugriffsversuche	225
8.4	Praxisbeispiel: Vokabeltrainer	226
8.5	Typische Fehler	228
8.6	Aufgaben	228
8.7	Lösungen	231
9	Ein- und Ausgabe	235
9.1	Files	235
9.1.1	Die Rolle der Files bei E/A-Operationen	235
9.1.2	Was ist ein File?	236

9.1.3	Ein File-Objekt erzeugen	237
9.1.4	Speichern einer Zeichenkette	238
9.1.5	Laden einer Zeichenkette aus einer Datei	239
9.1.6	Absolute und relative Pfade	239
9.1.7	Zwischenspeichern ohne zu schließen	242
9.1.8	Zugriff auf Files (lesen und schreiben)	242
9.1.9	Speichern beliebiger Daten auf Files	244
9.2	Mehr Zuverlässigkeit durch try- und with-Anweisungen	245
9.2.1	try...finally	246
9.2.2	With-Anweisungen	247
9.3	Objekte speichern mit pickle	248
9.3.1	Funktionen zum Speichern und Laden	249
9.4	Die Pseudofiles sys.stdin und sys.stdout	250
9.5	Ausgabe von Werten mit der print()-Funktion	251
9.5.1	Anwendung: Ausgabe von Tabellen	253
9.6	Kommandozeilen-Argumente (Optionen)	253
9.7	Übungen	256
9.8	Lösungen	259
10	Definition eigener Klassen	265
10.1	Klassen und Objekte	265
10.2	Definition von Klassen	267
10.3	Objekte (Instanzen)	269
10.4	Zugriff auf Attribute – Sichtbarkeit	272
10.4.1	Öffentliche Attribute	272
10.4.2	Private Attribute	273
10.4.3	Properties	275
10.4.4	Dynamische Erzeugung von Attributen	277
10.5	Methoden	277
10.5.1	Polymorphismus – Überladen von Operatoren	278
10.6	Statische Methoden	282
10.7	Abstraktion, Verkapselung und Geheimnisprinzip	283
10.8	Vererbung	284
10.8.1	Spezialisierungen	284
10.8.2	Beispiel: Die Klasse Konto – eine Spezialisierung der Klasse Geld	285
10.8.3	Standardklassen als Basisklassen	288

10.9	Hinweise zum Programmierstil	290
10.9.1	Bezeichner	290
10.9.2	Sichtbarkeit	290
10.9.3	Dokumentation von Klassen	292
10.10	Typische Fehler	292
10.11	Aufgaben	294
10.12	Lösungen	297
11	Klassenbibliotheken in Modulen speichern	303
11.1	Testen einer Klasse in einem lauffähigen Stand-alone-Skript	303
11.2	Module speichern und importieren	305
11.3	Den Zugang zu einem Modul sicherstellen	307
11.4	Programmierstil: Verwendung und Dokumentation von Modulen ...	309
12	Objektorientiertes Modellieren	311
12.1	Phasen einer objektorientierten Software-Entwicklung	311
12.2	Fallstudie: Modell eines Wörterbuchs	312
12.2.1	OOA: Entwicklung einer Klassenstruktur	312
12.2.2	OOD: Entwurf einer Klassenstruktur für eine Implementierung in Python	316
12.2.3	OOP: Implementierung der Klassenstruktur	318
12.3	Assoziationen zwischen Klassen	322
12.3.1	Reflexive Assoziationen	322
12.3.2	Aggregation	324
12.4	Beispiel: Management eines Musicals	325
12.4.1	OOA	325
12.4.2	OOD	327
12.4.3	OOP	327
12.5	Aufgaben	337
12.6	Lösungen	338
13	Verarbeitung von Zeichenketten	343
13.1	Standardmethoden zur Verarbeitung von Zeichenketten	343
13.1.1	Formatieren	344
13.1.2	Schreibweise	344
13.1.3	Tests	345
13.1.4	Entfernen und Aufspalten	346
13.1.5	Suchen und Ersetzen	347

13.2	Codierung und Decodierung	347
13.2.1	Platonische Zeichen und Unicode	347
13.2.2	Zeichenketten durch Bytefolgen darstellen	349
13.3	Automatische Textproduktion	351
13.3.1	Texte mit variablen Teilen – Anwendung der String-Methode format()	351
13.3.2	Eine Tabelle erstellen	354
13.3.3	Mahnbriefe	355
13.3.4	Textuelle Repräsentation eines Objektes	356
13.4	Analyse von Texten	358
13.4.1	Chat Bots	358
13.4.2	Textanalyse mit einfachen Vorkommenstests	359
13.5	Reguläre Ausdrücke	361
13.5.1	Aufbau eines regulären Ausdrucks	362
13.5.2	Objekte für reguläre Ausdrücke (RE-Objekte)	365
13.5.3	Analyse von Strings mit match() und search()	366
13.5.4	Textpassagen extrahieren mit findall()	367
13.5.5	Zeichenketten zerlegen mit split()	369
13.5.6	Teilstrings ersetzen mit sub()	370
13.5.7	Match-Objekte	370
13.6	Den Computer zum Sprechen bringen – Sprachsynthese	373
13.6.1	Den Klang der Stimme verändern	375
13.7	Aufgaben	378
13.8	Lösungen	381
14	Systemfunktionen	391
14.1	Das Modul sys – die Schnittstelle zum Laufzeitsystem	391
14.1.1	Informationen über die aktuelle Systemumgebung	392
14.1.2	Standardeingabe und -ausgabe	393
14.1.3	Die Objektverwaltung beobachten mit getrefcount()	394
14.1.4	Ausführung eines Skripts beenden	395
14.2	Das Modul os – die Schnittstelle zum Betriebssystem	395
14.2.1	Dateien und Verzeichnisse suchen	396
14.2.2	Zugriffrechte abfragen und ändern (Windows und Unix) ...	397
14.2.3	Dateien und Verzeichnisse anlegen und modifizieren	399
14.2.4	Merkmale von Dateien und Verzeichnissen abfragen	400
14.2.5	Pfade verarbeiten	401

14.2.6	Umgebungsvariablen	403
14.2.7	Systematisches Durchlaufen eines Verzeichnisbaumes	404
14.3	Datum und Zeit	406
14.3.1	Funktionen des Moduls time	407
14.3.2	Sekundenformat	408
14.3.3	Zeit-Tupel	408
14.3.4	Zeitstrings	409
14.3.5	Einen Prozess unterbrechen mit sleep()	410
14.4	Aufgaben	411
14.5	Lösungen	412
15	Gestaltung von grafischen Benutzungsoberflächen	417
15.1	Ein einführendes Beispiel	418
15.2	Einfache Widgets	421
15.3	Die Master-Slave-Hierarchie	422
15.4	Optionen der Widgets	423
15.4.1	Optionen bei der Instanziierung setzen	423
15.4.2	Widget-Optionen nachträglich konfigurieren	424
15.4.3	Fonts	425
15.4.4	Farben	426
15.4.5	Rahmen	427
15.4.6	Die Größe eines Widgets	427
15.4.7	Leerraum um Text	429
15.5	Gemeinsame Methoden der Widgets	430
15.6	Die Klasse Tk	431
15.7	Die Klasse Button	431
15.8	Die Klasse Label	432
15.8.1	Dynamische Konfiguration der Beschriftung	432
15.8.2	Verwendung von Kontrollvariablen	433
15.9	Die Klasse Entry	435
15.10	Die Klasse Radiobutton	437
15.11	Die Klasse Checkbutton	439
15.12	Die Klasse Scale	441
15.13	Die Klasse Frame	443
15.14	Aufgaben	443
15.15	Lösungen	445

16	Layout	451
16.1	Der Packer	451
16.2	Layout-Fehler	453
16.3	Raster-Layout	454
16.4	Vorgehensweise bei der GUI-Entwicklung	458
	16.4.1 Die Benutzungsoberfläche gestalten	461
	16.4.2 Funktionalität hinzufügen	464
16.5	Aufgaben	465
16.6	Lösungen	468
17	Grafik	479
17.1	Die tkinter-Klasse Canvas	479
	17.1.1 Generierung grafischer Elemente – ID, Positionierung und Display-Liste	480
	17.1.2 Grafische Elemente gestalten	482
	17.1.3 Visualisieren mit Kreisdiagrammen	484
17.2	Die Klasse PhotoImage	487
	17.2.1 Eine Pixelgrafik erzeugen	488
	17.2.2 Fotos analysieren und verändern	491
17.3	Bilder in eine Benutzungsoberfläche einbinden	493
	17.3.1 Icons auf Schaltflächen	493
	17.3.2 Hintergrundbilder	494
17.4	Aufgaben	497
17.5	Lösungen	498
18	Event-Verarbeitung	501
18.1	Einführendes Beispiel	502
18.2	Event-Sequenzen	504
	18.2.1 Event-Typen	504
	18.2.2 Qualifizierer für Maus- und Tastatur-Events	504
	18.2.3 Modifizierer	506
18.3	Beispiel: Tastaturereignisse verarbeiten	506
18.4	Programmierung eines Eventhandlers	508
	18.4.1 Beispiel für eine Event-Auswertung	509
18.5	Bindemethoden	510
18.6	Aufgaben	510
18.7	Lösungen	513

19	Komplexe Benutzungsoberflächen	519
19.1	Text-Widgets	519
19.1.1	Methoden der Text-Widgets	520
19.2	Rollbalken (Scrollbars)	522
19.3	Menüs	524
19.3.1	Die Klasse Menu	524
19.3.2	Methoden der Klasse Menu	525
19.4	Texteditor mit Menüleiste und Pulldown-Menü	526
19.5	Dialogboxen	528
19.6	Aufgaben	532
19.7	Lösungen	533
20	Threads	537
20.1	Funktionen in einem Thread ausführen	538
20.2	Thread-Objekte erzeugen – die Klasse Thread	540
20.3	Aufgaben	543
20.4	Lösungen	544
21	Fehler finden und vermeiden	549
21.1	Testen von Bedingungen	549
21.1.1	Ausnahmen (Exceptions)	549
21.1.2	Testen von Vor- und Nachbedingungen mit assert()	551
21.1.3	Debugging-Modus und optimierter Modus	553
21.1.4	Ausnahmen gezielt auslösen	555
21.2	Selbstdokumentation	556
21.3	Dokumentation eines Programmlaufs mit Log-Dateien	557
21.3.1	Grundfunktionen	558
21.3.2	Logging-Levels	558
21.3.3	Logger-Objekte	563
21.3.4	Das Format der Logging-Meldungen konfigurieren	564
21.4	Debugging	565
21.5	Aufgabe	567
21.6	Lösung	567
22	CGI-Programmierung	569
22.1	Wie funktionieren CGI-Skripte?	569
22.2	Aufbau eines einfachen CGI-Skripts	571
22.3	CGI-Skripte ausführen	573

22.4	Kommunikation über interaktive Webseiten	576
22.4.1	Aufbau eines HTML-Formulars	576
22.4.2	Eingabekomponenten in einem HTML-Formular	578
22.5	Verarbeitung von Eingabedaten in einem CGI-Skript	580
22.6	Sonderzeichen handhaben	582
22.7	CGI-Skripte debuggen	583
22.8	Objektorientierte CGI-Skripte – Beispiel: ein Chatroom	585
22.9	CGI-Skripte mit Cookies	591
22.10	Aufgaben	594
22.11	Lösungen	596
23	Internet-Programmierung	601
23.1	Was ist ein Protokoll?	601
23.2	Übertragung von Dateien mit FTP	602
23.2.1	Das Modul ftplib	603
23.2.2	Navigieren und Downloaden	604
23.2.3	Ein Suchroboter für FTP-Server	607
23.3	Zugriff auf Webseiten mit HTTP	611
23.3.1	Automatische Auswertung von Webseiten	612
23.4	E-Mails senden mit SMTP	614
23.5	Aufgaben	617
23.6	Lösungen	619
24	Datenbanken	627
24.1	Was ist ein Datenbanksystem?	627
24.2	Entity-Relationship-Diagramme (ER-Diagramme)	628
24.3	Relationale Datenbanken	629
24.4	Darstellung von Relationen als Listen oder Dictionaries	630
24.5	Das Modul sqlite3	631
24.5.1	Eine Tabelle anlegen	631
24.5.2	Anfragen an eine Datenbank	633
24.5.3	SQL-Anweisungen mit variablen Teilen	634
24.5.4	SQL-Injections	635
24.6	Online-Redaktionssystem mit Datenbankbindung	636
24.6.1	Objektorientierte Analyse (OOA)	638
24.6.2	Objektorientierter Entwurf des Systems (OOD)	638
24.6.3	Exkurs: Authentifizieren mit MD5-Fingerprints	640
24.6.4	Implementierung des Redaktionssystems mit Python (OOP)	641

24.7	Aufgaben	650
24.8	Lösungen	651
25	Fortgeschrittene Programmiertechniken – Testen und Tuning	655
25.1	Automatisiertes Testen	655
25.2	Testen mit Docstrings – das Modul doctest	656
25.3	Praxisbeispiel: Suche nach dem Wort des Jahres	658
25.4	Klassen testen mit doctest	665
25.4.1	Wie testet man eine Klasse?	665
25.4.2	Normalisierte Whitespaces – doctest-Direktiven	666
25.4.3	Ellipsen verwenden	666
25.4.4	Dictionaries testen	667
25.5	Gestaltung von Testreihen mit unittest	667
25.5.1	Einführendes Beispiel mit einem Testfall	667
25.5.2	Klassen des Moduls unittest	669
25.5.3	Weiterführendes Beispiel	671
25.6	Tuning	675
25.6.1	Performanzanalyse mit dem Profiler	675
25.6.2	Praxisbeispiel: Auswertung astronomischer Fotografien ...	677
25.6.3	Performanzanalyse und Tuning	683
25.7	Aufgaben	684
25.8	Lösungen	686
26	XML	693
26.1	Was ist XML?	693
26.2	XML-Dokumente	694
26.3	Ein XML-Dokument als Baum	696
26.4	DOM	697
26.5	Das Modul xml.dom.minidom	700
26.5.1	XML-Dokumente und DOM-Objekte	700
26.5.2	Die Basisklasse Node	702
26.5.3	Die Klassen Document, Element und Text	704
26.6	Attribute von XML-Elementen	706
26.7	Anwendungsbeispiel 1: Eine XML-basierte Klasse	706
26.8	Anwendungsbeispiel 2: Datenkommunikation mit XML	709
26.8.1	Überblick	710
26.8.2	Das Client-Programm	711
26.8.3	Das Server-Programm	714

26.9	Aufgaben	718
26.10	Lösungen	719
27	Modellieren mit Kellern, Schlangen und Graphen	721
27.1	Stack (Keller, Stapel)	721
27.2	Queue (Schlange)	724
27.3	Schlangen und das Producer-Consumer-Pattern – das Standardmodul queue	725
27.4	Graphen	731
27.5	Aufgaben	741
27.6	Lösungen	743
A	Anhang	747
A.1	Zeichencodierung	747
	A.1.1 Codierung von Sonderzeichen in HTML	747
	A.1.2 Oktettcodierung ISO-8859-1 (Unicode o-255)	747
A.2	Quellen im WWW	752
A.3	Standardfunktionen	752
A.4	Mathematische Funktionen	755
	A.4.1 Das Modul math	755
	A.4.2 Das Modul random	756
A.5	EBNF-Grammatik	757
B	Glossar	761
C	Inhalt der CD	771
	Stichwortverzeichnis	773