

# Inhaltsverzeichnis

	<b>Einleitung</b> .....	21
	Warum Python? .....	21
	Python 3 .....	21
	An wen wendet sich dieses Buch? .....	21
	Inhalt und Aufbau .....	22
	Hinweise zur Typographie .....	22
	Programmbeispiele .....	23
<b>I</b>	<b>Grundlagen</b> .....	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	Hintergrund: Geschichte der objektorientierten Programmierung .....	34
I.9	Aufgaben .....	35
I.10	Lösungen .....	36
<b>2</b>	<b>Der Einstieg – Python im interaktiven Modus</b> .....	37
2.1	Python installieren .....	37
2.2	Python im interaktiven Modus .....	40
2.2.1	Start des Python-Interpreters in einem Konsole-Fenster ...	40
2.2.2	Die Python-Shell von IDLE .....	40
2.2.3	Die ersten Python-Befehle ausprobieren .....	41
2.2.4	Hotkeys .....	41
2.3	Objekte .....	42

2.4	Namen .....	44
2.5	Hintergrund: Syntax-Regeln für Bezeichner .....	45
2.6	Schlüsselwörter .....	46
2.7	Anweisungen .....	46
2.7.1	Ausdrucksanweisungen .....	46
2.7.2	Import-Anweisungen .....	52
2.7.3	Zuweisungen .....	53
2.7.4	Erweiterte Zuweisungen .....	56
2.7.5	Hintergrund: Dynamische Typisierung .....	56
2.8	Aufgaben .....	57
2.9	Lösungen .....	59
3	<b>Python-Skripte</b> .....	61
3.1	Skripte editieren und ausführen mit IDLE .....	61
3.2	Ausführen eines Python-Skripts .....	62
3.3	Kommentare .....	65
3.4	Die Zeilenstruktur von Python-Programmen .....	65
3.5	Das EVA-Prinzip .....	69
3.6	Phasen der Programmentwicklung .....	70
3.7	Guter Programmierstil .....	71
3.8	Die Kunst des Fehlerfindens .....	74
3.9	Aufgaben .....	76
3.10	Lösungen .....	77
4	<b>Standard-Datentypen</b> .....	79
4.1	Daten als Objekte .....	79
4.2	Fundamentale Datentypen im Überblick .....	81
4.3	Typen und Klassen .....	82
4.4	NoneType .....	83
4.5	Wahrheitswerte – der Datentyp bool .....	83
4.6	Ganze Zahlen .....	84
4.7	Gleitkommazahlen .....	86
4.8	Komplexe Zahlen .....	87
4.9	Arithmetische Operatoren für Zahlen .....	88
4.10	Sequenzen .....	93
4.10.1	Zeichenketten (Strings) .....	94
4.10.2	Bytestrings .....	96

4.10.3	Tupel .....	97
4.10.4	Liste .....	98
4.10.5	Bytearray .....	99
4.10.6	Einige Grundoperationen für Sequenzen .....	99
4.10.7	Veränderbare und unveränderbare Sequenzen .....	102
4.11	Mengen .....	103
4.12	Dictionaries .....	104
4.13	Typumwandlungen .....	104
4.13.1	int() .....	105
4.13.2	float() .....	106
4.13.3	complex() .....	107
4.13.4	bool() .....	107
4.13.5	str() .....	107
4.13.6	dict(), list() und tuple() .....	108
4.14	Aufgaben .....	108
4.15	Lösungen .....	111
5	<b>Kontrollstrukturen</b> .....	115
5.1	Einfache Bedingungen .....	115
5.1.1	Vergleiche .....	115
5.1.2	Zugehörigkeit zu einer Menge (in, not in) .....	119
5.1.3	Beliebige Ausdrücke als Bedingungen .....	119
5.2	Zusammengesetzte Bedingungen – logische Operatoren .....	120
5.2.1	Negation (not) .....	120
5.2.2	Konjunktion (and) .....	121
5.2.3	Disjunktion (or) .....	122
5.2.4	Formalisierung von Bedingungen .....	123
5.2.5	Hinweis zum Programmierstil .....	124
5.3	Programmverzweigungen (bedingte Anweisungen) .....	124
5.3.1	Einseitige Verzweigung (if) .....	125
5.3.2	Zweiseitige Verzweigung (if-else) .....	125
5.3.3	Mehrfache Fallunterscheidung (elif) .....	126
5.3.4	Bedingte Ausdrücke .....	128
5.4	Bedingte Wiederholung (while) .....	128
5.4.1	Endlosschleifen .....	129
5.5	Iteration über eine Kollektion (for) .....	131
5.5.1	Zählschleifen – Verwendung von range() .....	132

5.5.2	Verschachtelte Iterationen .....	133
5.5.3	Vertiefung: Iterative Berechnung rekursiver Folgen .....	135
5.6	Abbruch einer Schleife mit break .....	135
5.6.1	Abbruch eines Schleifendurchlaufs mit continue .....	136
5.7	Abfangen von Ausnahmen mit try .....	137
5.7.1	try...except .....	138
5.8	Aufgaben .....	140
5.9	Lösungen .....	144
<b>6</b>	<b>Funktionen .....</b>	<b>149</b>
6.1	Aufruf von Funktionen .....	149
6.2	Definition von Funktionen .....	152
6.3	Schrittweise Verfeinerung .....	154
6.4	Ausführung von Funktionen .....	157
6.4.1	Globale und lokale Namen .....	157
6.4.2	Seiteneffekte – die global-Anweisung .....	161
6.4.3	Parameterübergabe .....	162
6.5	Voreingestellte Parameterwerte .....	163
6.5.1	Schlüsselwort-Argumente .....	166
6.6	Funktionen mit beliebiger Anzahl von Parametern .....	167
6.7	Lokale Funktionen .....	168
6.8	Rekursive Funktionen .....	170
6.9	Experimente zur Rekursion mit der Turtle-Grafik .....	171
6.9.1	Turtle-Befehle im interaktiven Modus .....	171
6.9.2	Eine rekursive Spirale .....	173
6.9.3	Baumstrukturen .....	174
6.9.4	Künstlicher Blumenkohl – selbstähnliche Bilder .....	176
6.10	Rekursive Zahlenfunktionen .....	177
6.11	Hintergrund: Wie werden rekursive Funktionen ausgeführt? .....	178
6.11.1	Execution Frames .....	178
6.11.2	Rekursionstiefe .....	179
6.12	Funktionen als Objekte .....	181
6.13	Hintergrund: Lambda-Formen .....	182
6.14	Hinweise zum Programmierstil .....	183
6.14.1	Allgemeines .....	183
6.14.2	Funktionsnamen .....	183
6.14.3	Kommentierte Parameter .....	184
6.14.4	Docstrings .....	184

6.15	Aufgaben .....	185
6.15.1	Aufgabe 5 .....	188
6.16	Lösung .....	188
7	<b>Sequenzen, Mengen und Generatoren .....</b>	<b>193</b>
7.1	Gemeinsame Operationen für Sequenzen .....	193
7.1.1	Zugriff auf Elemente einer Sequenz .....	194
7.1.2	Slicing von Sequenzen .....	195
7.2	Vertiefung: Rekursive Funktionen für Sequenzen .....	196
7.2.1	Rekursives Summieren .....	196
7.2.2	Rekursive Suche .....	196
7.3	Tupel .....	198
7.4	Listen .....	199
7.4.1	Eine Liste erzeugen .....	200
7.4.2	Eine Liste verändern .....	202
7.4.3	Flache und tiefe Kopien .....	204
7.4.4	Listen sortieren .....	205
7.4.5	Binäre Suche in einer sortierten Liste .....	207
7.4.6	Zwei Sortierverfahren im Vergleich .....	208
7.4.7	Modellieren mit Listen – Beispiel: die Charts .....	212
7.5	Generatoren .....	216
7.5.1	Generatorausdrücke .....	217
7.5.2	Generatorfunktionen .....	217
7.5.3	Iteratoren .....	219
7.5.4	Verwendung von Generatoren .....	220
7.6	Mengen .....	221
7.6.1	Operationen für Mengen .....	222
7.6.2	Modellieren mit Mengen – Beispiel: Graphen .....	223
7.7	Aufgaben .....	226
7.8	Lösungen .....	228
8	<b>Dictionaries .....</b>	<b>231</b>
8.1	Operationen für Dictionaries .....	231
8.2	Wie erstellt man ein Dictionary? .....	232
8.2.1	Definition mit einem Dictionary-Display .....	232
8.2.2	Schrittweiser Aufbau eines Dictionarys .....	234
8.2.3	Ein Dictionary aus anderen Dictionaries zusammensetzen – update() .....	235

8.3	Zugriff auf Daten in einem Dictionary .....	235
8.3.1	Vergebliche Zugriffsversuche .....	235
8.4	Praxisbeispiel: Vokabeltrainer .....	236
8.5	Typische Fehler .....	238
8.6	Aufgaben .....	238
8.7	Lösungen .....	241
9	<b>Ein- und Ausgabe</b> .....	245
9.1	<b>Files</b> .....	245
9.1.1	Die Rolle der Files bei E/A-Operationen .....	245
9.1.2	Was ist ein File? .....	246
9.1.3	Ein File-Objekt erzeugen .....	247
9.1.4	Speichern einer Zeichenkette .....	248
9.1.5	Laden einer Zeichenkette aus einer Datei .....	249
9.1.6	Absolute und relative Pfade .....	249
9.1.7	Zwischenspeichern, ohne zu schließen .....	252
9.1.8	Zugriff auf Files (lesen und schreiben) .....	252
9.1.9	Speichern beliebiger Daten auf Files .....	254
9.2	Mehr Zuverlässigkeit durch try- und with-Anweisungen .....	255
9.2.1	try...finally .....	256
9.2.2	With-Anweisungen .....	257
9.3	Objekte speichern mit pickle .....	258
9.3.1	Funktionen zum Speichern und Laden .....	259
9.4	Die Pseudofiles sys.stdin und sys.stdout .....	260
9.5	Ausgabe von Werten mit der print()-Funktion .....	261
9.5.1	Anwendung: Ausgabe von Tabellen .....	263
9.6	Kommandozeilen-Argumente (Optionen) .....	263
9.7	Übungen .....	266
9.8	Lösungen .....	269
10	<b>Definition eigener Klassen</b> .....	275
10.1	Klassen und Objekte .....	275
10.2	Definition von Klassen .....	277
10.3	Objekte (Instanzen) .....	279
10.4	Zugriff auf Attribute – Sichtbarkeit .....	282
10.4.1	Öffentliche Attribute .....	282
10.4.2	Private Attribute .....	283
10.4.3	Properties .....	285
10.4.4	Dynamische Erzeugung von Attributen .....	287

10.5	Methoden .....	287
10.5.1	Polymorphismus – Überladen von Operatoren .....	288
10.6	Statische Methoden .....	292
10.7	Abstraktion, Verkapselung und Geheimnisprinzip .....	293
10.8	Vererbung .....	294
10.8.1	Spezialisierungen .....	294
10.8.2	Beispiel: Die Klasse Konto – eine Spezialisierung der Klasse Geld .....	295
10.8.3	Vertiefung: Standardklassen als Basisklassen .....	298
10.9	Hinweise zum Programmierstil .....	300
10.9.1	Bezeichner .....	300
10.9.2	Sichtbarkeit .....	300
10.9.3	Dokumentation von Klassen .....	302
10.10	Typische Fehler .....	302
10.11	Aufgaben .....	304
10.12	Lösungen .....	307
II	<b>Klassenbibliotheken in Modulen speichern .....</b>	<b>313</b>
II.1	Testen einer Klasse in einem lauffähigen Stand-alone-Skript .....	313
II.2	Module speichern und importieren .....	315
II.3	Den Zugang zu einem Modul sicherstellen .....	317
II.4	Programmierstil: Verwendung und Dokumentation von Modulen ..	319
12	<b>Objektorientiertes Modellieren .....</b>	<b>321</b>
12.1	Phasen einer objektorientierten Software-Entwicklung .....	321
12.2	Fallstudie: Modell eines Wörterbuchs .....	322
12.2.1	OOA: Entwicklung einer Klassenstruktur .....	322
12.2.2	OOD: Entwurf einer Klassenstruktur für eine Implementierung in Python .....	326
12.2.3	OOP: Implementierung der Klassenstruktur .....	328
12.3	Assoziationen zwischen Klassen .....	332
12.3.1	Reflexive Assoziationen .....	332
12.3.2	Aggregation .....	334
12.4	Beispiel: Management eines Musicals .....	335
12.4.1	OOA .....	335
12.4.2	OOD .....	337
12.4.3	OOP .....	337
12.5	Aufgaben .....	347
12.6	Lösungen .....	348

<b>13</b>	<b>Verarbeitung von Zeichenketten</b>	<b>353</b>
13.1	Standardmethoden zur Verarbeitung von Zeichenketten	353
13.1.1	Formatieren	354
13.1.2	Schreibweise	354
13.1.3	Tests	355
13.1.4	Entfernen und Aufspalten	356
13.1.5	Suchen und Ersetzen	357
13.2	Codierung und Decodierung	357
13.2.1	Platonische Zeichen und Unicode	357
13.2.2	Vertiefung: Zeichenketten durch Bytefolgen darstellen	359
13.3	Automatische Textproduktion	361
13.3.1	Texte mit variablen Teilen – Anwendung der String-Methode format()	361
13.3.2	Vertiefung: Eine Tabelle erstellen	364
13.3.3	Mahnbriefe	365
13.3.4	Textuelle Repräsentation eines Objektes	366
13.4	Analyse von Texten	368
13.4.1	Chat Bots	368
13.4.2	Textanalyse mit einfachen Vorkommenstests	369
13.5	Reguläre Ausdrücke	371
13.5.1	Aufbau eines regulären Ausdrucks	372
13.5.2	Objekte für reguläre Ausdrücke (RE-Objekte)	375
13.5.3	Analyse von Strings mit match() und search()	376
13.5.4	Textpassagen extrahieren mit findall()	377
13.5.5	Zeichenketten zerlegen mit split()	379
13.5.6	Teilstrings ersetzen mit sub()	380
13.5.7	Match-Objekte	380
13.6	Den Computer zum Sprechen bringen – Sprachsynthese	383
13.6.1	Den Klang der Stimme verändern	386
13.7	Aufgaben	388
13.8	Lösungen	391
<b>14</b>	<b>Systemfunktionen</b>	<b>401</b>
14.1	Das Modul sys – die Schnittstelle zum Laufzeitsystem	401
14.1.1	Informationen über die aktuelle Systemumgebung	402
14.1.2	Standardeingabe und -ausgabe	403
14.1.3	Die Objektverwaltung beobachten mit getrefcount()	404
14.1.4	Ausführung eines Skripts beenden	405



14.2	Das Modul os – die Schnittstelle zum Betriebssystem .....	405
14.2.1	Dateien und Verzeichnisse suchen .....	406
14.2.2	Hintergrund: Zugriffsrechte abfragen und ändern (Windows und Unix) .....	407
14.2.3	Dateien und Verzeichnisse anlegen und modifizieren .....	409
14.2.4	Merkmale von Dateien und Verzeichnissen abfragen .....	410
14.2.5	Pfade verarbeiten .....	411
14.2.6	Hintergrund: Umgebungsvariablen .....	413
14.2.7	Systematisches Durchlaufen eines Verzeichnisbaumes ....	414
14.3	Datum und Zeit .....	416
14.3.1	Funktionen des Moduls time .....	417
14.3.2	Sekundenformat .....	418
14.3.3	Zeit-Tupel .....	418
14.3.4	Zeitstrings .....	419
14.3.5	Einen Prozess unterbrechen mit sleep() .....	420
14.4	Aufgaben .....	421
14.5	Lösungen .....	422
15	<b>Gestaltung von grafischen Benutzungsoberflächen .....</b>	<b>427</b>
15.1	Ein einführendes Beispiel .....	428
15.2	Einfache Widgets .....	431
15.3	Die Master-Slave-Hierarchie .....	432
15.4	Optionen der Widgets .....	433
15.4.1	Optionen bei der Instanziierung setzen .....	433
15.4.2	Widget-Optionen nachträglich konfigurieren .....	434
15.4.3	Fonts .....	435
15.4.4	Farben .....	436
15.4.5	Rahmen .....	437
15.4.6	Die Größe eines Widgets .....	437
15.4.7	Leerraum um Text .....	439
15.5	Gemeinsame Methoden der Widgets .....	440
15.6	Die Klasse Tk .....	441
15.7	Die Klasse Button .....	441
15.8	Die Klasse Label .....	442
15.8.1	Dynamische Konfiguration der Beschriftung .....	442
15.8.2	Verwendung von Kontrollvariablen .....	443
15.9	Die Klasse Entry .....	445
15.10	Die Klasse Radiobutton .....	447

15.11	Die Klasse Checkbutton .....	449
15.12	Die Klasse Scale .....	451
15.13	Die Klasse Frame .....	453
15.14	Aufgaben .....	453
15.15	Lösungen .....	455
16	<b>Layout</b> .....	461
16.1	Der Packer .....	461
16.2	Layout-Fehler .....	463
16.3	Raster-Layout .....	464
16.4	Vorgehensweise bei der GUI-Entwicklung .....	468
16.4.1	Die Benutzungsoberfläche gestalten .....	471
16.4.2	Funktionalität hinzufügen .....	474
16.5	Aufgaben .....	475
16.6	Lösungen .....	478
17	<b>Grafik</b> .....	489
17.1	Die tkinter-Klasse Canvas .....	489
17.1.1	Generierung grafischer Elemente – ID, Positionierung und Display-Liste .....	490
17.1.2	Grafische Elemente gestalten .....	492
17.1.3	Visualisieren mit Kreisdiagrammen .....	494
17.2	Die Klasse PhotoImage .....	497
17.2.1	Eine Pixelgrafik erzeugen .....	498
17.2.2	Fotos analysieren und verändern .....	501
17.3	Bilder in eine Benutzungsoberfläche einbinden .....	503
17.3.1	Icons auf Schaltflächen .....	503
17.3.2	Hintergrundbilder .....	504
17.3.3	Hintergrund: Das PPM-Format .....	507
17.3.4	Steganographie – Informationen in Bildern verstecken ...	508
17.4	Aufgaben .....	510
17.5	Lösungen .....	511
18	<b>Event-Verarbeitung</b> .....	515
18.1	Einführendes Beispiel .....	516
18.2	Event-Sequenzen .....	518
18.2.1	Event-Typen .....	518
18.2.2	Qualifizierer für Maus- und Tastatur-Events .....	518
18.2.3	Modifizierer .....	520

18.3	Beispiel: Tastaturereignisse verarbeiten .....	520
18.4	Programmierung eines Eventhandlers .....	522
18.4.1	Beispiel für eine Event-Auswertung .....	523
18.5	Bindemethoden .....	524
18.6	Aufgaben .....	524
18.7	Lösungen .....	527
19	<b>Komplexe Benutzungsoberflächen</b> .....	533
19.1	Text-Widgets .....	533
19.1.1	Methoden der Text-Widgets .....	534
19.2	Rollbalken (Scrollbars) .....	536
19.3	Menüs .....	538
19.3.1	Die Klasse Menu .....	538
19.3.2	Methoden der Klasse Menu .....	539
19.4	Texteditor mit Menüleiste und Pulldown-Menü .....	540
19.5	Dialogboxen .....	542
19.6	Applikationen mit mehreren Fenstern .....	546
19.7	Aufgaben .....	549
19.8	Lösungen .....	550
20	<b>Threads</b> .....	555
20.1	Funktionen in einem Thread ausführen .....	556
20.2	Thread-Objekte erzeugen – die Klasse Thread .....	558
20.3	Aufgaben .....	561
20.4	Lösungen .....	562
21	<b>Fehler finden und vermeiden</b> .....	567
21.1	Testen von Bedingungen .....	567
21.1.1	Ausnahmen (Exceptions) .....	567
21.1.2	Testen von Vor- und Nachbedingungen mit assert .....	568
21.1.3	Vertiefung: Programmabstürze ohne Fehlermeldung .....	571
21.2	Debugging-Modus und optimierter Modus .....	573
21.3	Ausnahmen gezielt auslösen .....	574
21.4	Selbstdokumentation .....	575
21.5	Dokumentation eines Programmlaufs mit Log-Dateien .....	577
21.5.1	Grundfunktionen .....	577
21.5.2	Beispiel: Logging in der GUI-Programmierung .....	578

21.6	Vertiefung: Professionelles Arbeiten mit Logging .....	579
21.6.1	Logging-Levels .....	579
21.6.2	Logger-Objekte .....	584
21.6.3	Das Format der Logging-Meldungen konfigurieren .....	584
21.7	Debugging .....	586
21.8	Aufgabe .....	587
21.9	Lösung .....	588
22	<b>CGI-Programmierung</b> .....	589
22.1	Wie funktionieren CGI-Skripte? .....	589
22.2	Wie spät ist es? Aufbau eines CGI-Skripts .....	591
22.2.1	Ein einfacher HTTP-Server .....	594
22.2.2	Hintergrund: CGI-Skripte auf einem Host im Internet installieren .....	595
22.3	Kommunikation über interaktive Webseiten .....	596
22.3.1	Aufbau eines HTML-Formulars .....	597
22.3.2	Eingabekomponenten in einem HTML-Formular .....	598
22.4	Verarbeitung von Eingabedaten in einem CGI-Skript .....	600
22.5	Sonderzeichen handhaben .....	602
22.6	CGI-Skripte debuggen .....	604
22.7	Objektorientierte CGI-Skripte – Beispiel: ein Chatroom .....	605
22.8	CGI-Skripte mit Cookies .....	610
22.9	Aufgaben .....	613
22.10	Lösungen .....	615
23	<b>Internet-Programmierung</b> .....	621
23.1	Was ist ein Protokoll? .....	621
23.2	Übertragung von Dateien mit FTP .....	622
23.2.1	Das Modul ftplib .....	623
23.2.2	Navigieren und Downloaden .....	624
23.2.3	Ein Suchroboter für FTP-Server .....	626
23.3	Zugriff auf Webseiten mit HTTP .....	630
23.3.1	Automatische Auswertung von Webseiten .....	632
23.4	E-Mails senden mit SMTP .....	634
23.5	Aufgaben .....	637
23.6	Lösungen .....	639

<b>24</b>	<b>Datenbanken</b>	<b>645</b>
24.1	Was ist ein Datenbanksystem?	645
24.2	Entity-Relationship-Diagramme (ER-Diagramme)	646
24.3	Relationale Datenbanken	647
24.4	Darstellung von Relationen als Listen oder Dictionaries	648
24.5	Das Modul sqlite3	649
24.5.1	Eine Tabelle anlegen	649
24.5.2	Anfragen an eine Datenbank	651
24.5.3	SQL-Anweisungen mit variablen Teilen	652
24.5.4	SQL-Injections	653
24.6	Online-Redaktionssystem mit Datenbankanbindung	654
24.6.1	Objektorientierte Analyse (OOA)	656
24.6.2	Objektorientierter Entwurf des Systems (OOD)	656
24.6.3	Hintergrund: Authentifizieren mit MD5-Fingerprints	658
24.6.4	Implementierung des Redaktionssystems mit Python (OOP)	659
24.7	Aufgaben	668
24.8	Lösungen	669
<b>25</b>	<b>Fortgeschrittene Programmiertechniken – Testen und Tuning</b>	<b>673</b>
25.1	Automatisiertes Testen	673
25.2	Testen mit Docstrings – das Modul doctest	674
25.3	Praxisbeispiel: Suche nach dem Wort des Jahres	676
25.4	Klassen testen mit doctest	683
25.4.1	Wie testet man eine Klasse?	683
25.4.2	Normalisierte Whitespaces – doctest-Direktiven	684
25.4.3	Ellipsen verwenden	684
25.4.4	Dictionaries testen	685
25.5	Gestaltung von Testreihen mit unittest	685
25.5.1	Einführendes Beispiel mit einem Testfall	685
25.5.2	Klassen des Moduls unittest	687
25.5.3	Weiterführendes Beispiel	689
25.6	Tuning	693
25.6.1	Performanceanalyse mit dem Profiler	693
25.6.2	Praxisbeispiel: Auswertung astronomischer Fotografien	695
25.6.3	Performanceanalyse und Tuning	701
25.7	Aufgaben	702
25.8	Lösungen	704

<b>26</b>	<b>XML</b>	<b>711</b>
26.1	Was ist XML?	711
26.2	XML-Dokumente	712
26.3	Ein XML-Dokument als Baum	714
26.4	DOM	715
26.5	Das Modul <code>xml.dom.minidom</code>	718
26.5.1	XML-Dokumente und DOM-Objekte	718
26.5.2	Die Basisklasse <code>Node</code>	720
26.5.3	Die Klassen <code>Document</code> , <code>Element</code> und <code>Text</code>	722
26.6	Attribute von XML-Elementen	724
26.7	Anwendungsbeispiel 1: Eine XML-basierte Klasse	724
26.8	Anwendungsbeispiel 2: Datenkommunikation mit XML	727
26.8.1	Überblick	728
26.8.2	Das Client-Programm	729
26.8.3	Das Server-Programm	732
26.9	Aufgaben	736
26.10	Lösungen	737
<b>27</b>	<b>Modellieren mit Kellern, Schlangen und Graphen</b>	<b>739</b>
27.1	Stack (Keller, Stapel)	739
27.2	Queue (Schlange)	742
27.3	Schlangen und das Producer-Consumer-Pattern – das Standardmodul <code>queue</code>	743
27.4	Graphen	749
27.5	Aufgaben	759
27.6	Lösungen	761
<b>A</b>	<b>Anhang</b>	<b>765</b>
A.1	Zeichencodierung	765
A.1.1	Codierung von Sonderzeichen in HTML	765
A.1.2	Oktettcodierung ISO-8859-1 (Unicode 0-255)	765
A.2	Quellen im WWW	770
A.3	Standardfunktionen	770
A.4	Mathematische Funktionen	773
A.4.1	Das Modul <code>math</code>	773
A.4.2	Das Modul <code>random</code>	774
A.5	EBNF-Grammatik	775

<b>B</b>	<b>Glossar .....</b>	<b>779</b>
<b>C</b>	<b>Inhalt der CD .....</b>	<b>791</b>
	<b>Stichwortverzeichnis .....</b>	<b>793</b>