

# 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 .....	23
	Programmbeispiele .....	24
<b>1</b>	<b>Grundlagen</b> .....	<b>25</b>
1.1	Was ist Programmieren? .....	25
1.2	Hardware und Software .....	26
1.3	Programm als Algorithmus .....	27
1.4	Syntax und Semantik .....	28
1.5	Interpreter und Compiler .....	28
1.6	Programmierparadigmen .....	30
1.7	Objektorientierte Programmierung .....	31
	1.7.1 Strukturelle Zerlegung .....	31
	1.7.2 Die Welt als System von Objekten .....	32
	1.7.3 Objekte besitzen Attribute und beherrschen Methoden .....	33
	1.7.4 Objekte sind Instanzen von Klassen .....	34
1.8	Hintergrund: Geschichte der objektorientierten Programmierung .....	34
1.9	Aufgaben .....	35
1.10	Lösungen .....	36
<b>2</b>	<b>Der Einstieg – Python im interaktiven Modus</b> .....	<b>37</b>
2.1	Python installieren .....	37
2.2	Python im interaktiven Modus .....	40
	2.2.1 Start des Python-Interpreters in einem Konsolenfenster .....	40
	2.2.2 Die Python-Shell von IDLE .....	41
	2.2.3 Die ersten Python-Befehle ausprobieren .....	41
	2.2.4 Hotkeys .....	42
2.3	Objekte .....	43
2.4	Namen .....	44
2.5	Hintergrund: Syntax-Regeln für Bezeichner .....	45
2.6	Schlüsselwörter .....	46
2.7	Anweisungen .....	46
	2.7.1 Ausdruckanweisungen .....	47
	2.7.2 Import-Anweisungen .....	52
	2.7.3 Zuweisungen .....	53
	2.7.4 Erweiterte Zuweisungen .....	57

2.7.5	Hintergrund: Dynamische Typisierung .....	57
2.8	Aufgaben .....	58
2.9	Lösungen .....	60
<b>3</b>	<b>Python-Skripte</b> .....	<b>63</b>
3.1	Ausprobieren, nachmachen, besser machen! .....	63
3.2	Skripte editieren und ausführen mit IDLE .....	63
3.3	Ausführen eines Python-Skripts .....	65
3.4	Kommentare .....	67
3.5	Die Zeilenstruktur von Python-Programmen .....	68
3.6	Das EVA-Prinzip .....	71
3.7	Phasen der Programmentwicklung .....	73
3.8	Guter Programmierstil .....	74
3.9	Hintergrund: Die Kunst des Fehlerfindens .....	76
3.10	Weitere Entwicklungsumgebungen für Python .....	78
3.11	Aufgaben .....	79
3.12	Lösungen .....	80
<b>4</b>	<b>Standard-Datentypen</b> .....	<b>83</b>
4.1	Daten als Objekte .....	83
4.2	Fundamentale Datentypen im Überblick .....	85
4.3	Typen und Klassen .....	86
4.4	NoneType .....	87
4.5	Wahrheitswerte – der Datentyp bool .....	87
4.6	Ganze Zahlen .....	88
4.7	Gleitkommazahlen .....	90
4.8	Komplexe Zahlen .....	91
4.9	Arithmetische Operatoren für Zahlen .....	92
4.10	Sequenzen .....	97
4.10.1	Zeichenketten (Strings) .....	98
4.10.2	Bytestrings .....	100
4.10.3	Tupel .....	101
4.10.4	Liste .....	102
4.10.5	Bytearray .....	103
4.10.6	Einige Grundoperationen für Sequenzen .....	103
4.10.7	Veränderbare und unveränderbare Sequenzen .....	105
4.11	Mengen .....	106
4.12	Dictionaries .....	107
4.13	Typumwandlungen .....	107
4.13.1	int() .....	108
4.13.2	float() .....	109
4.13.3	complex() .....	110
4.13.4	bool() .....	110
4.13.5	str() .....	110
4.13.6	dict(), list() und tuple() .....	111

4.14	Aufgaben .....	111
4.15	Lösungen .....	114
<b>5</b>	<b>Kontrollstrukturen</b> .....	<b>117</b>
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 .....	126
5.3	Programmverzweigungen (bedingte Anweisungen) .....	126
5.3.1	Einseitige Verzweigung (if) .....	127
5.3.2	Zweiseitige Verzweigung (if-else) .....	127
5.3.3	Mehrfache Fallunterscheidung (elif) .....	128
5.3.4	Bedingte Ausdrücke .....	130
5.4	Bedingte Wiederholung (while) .....	130
5.4.1	Endlosschleifen .....	131
5.5	Iteration über eine Kollektion (for) .....	133
5.5.1	Zählschleifen – Verwendung von range() .....	134
5.5.2	Verschachtelte Iterationen .....	135
5.5.3	Vertiefung: Iterative Berechnung rekursiver Folgen .....	137
5.6	Abbruch einer Schleife mit break .....	137
5.6.1	Abbruch eines Schleifendurchlaufs mit continue .....	138
5.7	Abfangen von Ausnahmen mit try .....	139
5.7.1	try...except .....	140
5.8	Aufgaben .....	142
5.9	Lösungen .....	146
<b>6</b>	<b>Funktionen</b> .....	<b>151</b>
6.1	Aufruf von Funktionen .....	151
6.2	Definition von Funktionen .....	154
6.3	Schrittweise Verfeinerung .....	156
6.4	Ausführung von Funktionen .....	160
6.4.1	Globale und lokale Namen .....	160
6.4.2	Seiteneffekte – die global-Anweisung .....	163
6.4.3	Parameterübergabe .....	164
6.5	Voreingestellte Parameterwerte .....	166
6.5.1	Schlüsselwort-Argumente .....	168
6.6	Funktionen mit beliebiger Anzahl von Parametern .....	170
6.7	Lokale Funktionen .....	171
6.8	Rekursive Funktionen .....	172

6.9	Experimente zur Rekursion mit der Turtle-Grafik . . . . .	174
6.9.1	Turtle-Befehle im interaktiven Modus . . . . .	174
6.9.2	Eine rekursive Spirale. . . . .	175
6.9.3	Baumstrukturen . . . . .	177
6.9.4	Künstlicher Blumenkohl – selbstähnliche Bilder. . . . .	178
6.10	Rekursive Zahlenfunktionen . . . . .	180
6.11	Hintergrund: Wie werden rekursive Funktionen ausgeführt? . . . . .	181
6.11.1	Execution Frames . . . . .	181
6.11.2	Rekursionstiefe . . . . .	182
6.12	Funktionen als Objekte. . . . .	184
6.12.1	Hintergrund: Typen sind keine Funktionen . . . . .	185
6.13	Lambda-Formen . . . . .	185
6.14	Funktionsannotationen: Typen zuordnen . . . . .	186
6.15	Hinweise zum Programmierstil. . . . .	187
6.15.1	Allgemeines. . . . .	187
6.15.2	Funktionsnamen. . . . .	187
6.15.3	Kommentierte Parameter. . . . .	188
6.15.4	Docstrings . . . . .	188
6.16	Aufgaben . . . . .	189
6.17	Lösungen . . . . .	192
<b>7</b>	<b>Sequenzen, Mengen und Generatoren . . . . .</b>	<b>197</b>
7.1	Gemeinsame Operationen für Sequenzen . . . . .	197
7.1.1	Zugriff auf Elemente einer Sequenz. . . . .	198
7.1.2	Slicing von Sequenzen . . . . .	199
7.1.3	Auspacken (unpacking) . . . . .	200
7.2	Vertiefung: Rekursive Funktionen für Sequenzen . . . . .	201
7.2.1	Rekursives Summieren . . . . .	201
7.2.2	Rekursive Suche . . . . .	201
7.3	Tupel. . . . .	203
7.4	Listen . . . . .	204
7.4.1	Eine Liste erzeugen. . . . .	204
7.4.2	Eine Liste verändern . . . . .	207
7.4.3	Flache und tiefe Kopien . . . . .	209
7.4.4	Listen sortieren . . . . .	210
7.4.5	Binäre Suche in einer sortierten Liste. . . . .	212
7.4.6	Zwei Sortierverfahren im Vergleich . . . . .	213
7.4.7	Modellieren mit Listen – Beispiel: die Charts . . . . .	217
7.5	Generatoren . . . . .	221
7.5.1	Generatorausdrücke . . . . .	222
7.5.2	Generatorfunktionen . . . . .	222
7.5.3	Iteratoren. . . . .	224
7.5.4	Verwendung von Generatoren. . . . .	225
7.6	Mengen . . . . .	225
7.6.1	Operationen für Mengen . . . . .	227

7.6.2	Modellieren mit Mengen – Beispiel: Graphen .....	228
7.7	Aufgaben .....	231
7.8	Lösungen .....	233
<b>8</b>	<b>Dictionaries</b> .....	<b>235</b>
8.1	Operationen für Dictionaries .....	235
8.2	Wie erstellt man ein Dictionary? .....	236
8.2.1	Definition mit einem Dictionary-Display .....	236
8.2.2	Schrittweiser Aufbau eines Dictionarys. ....	238
8.2.3	Ein Dictionary aus anderen Dictionaries zusammensetzen – update() .....	239
8.3	Zugriff auf Daten in einem Dictionary .....	239
8.3.1	Vergebliche Zugriffsversuche .....	239
8.4	Praxisbeispiel: Vokabeltrainer. ....	240
8.5	Typische Fehler .....	242
8.6	Aufgaben .....	242
8.7	Lösungen .....	245
<b>9</b>	<b>Ein- und Ausgabe</b> .....	<b>249</b>
9.1	Files .....	249
9.1.1	Die Rolle der Files bei E/A-Operationen .....	249
9.1.2	Was ist ein File? .....	250
9.1.3	Ein File-Objekt erzeugen .....	251
9.1.4	Speichern einer Zeichenkette. ....	252
9.1.5	Laden einer Zeichenkette aus einer Datei. ....	253
9.1.6	Absolute und relative Pfade .....	253
9.1.7	Zwischenspeichern, ohne zu schließen .....	255
9.1.8	Zugriff auf Files (lesen und schreiben) .....	256
9.1.9	Speichern beliebiger Daten auf Files .....	258
9.2	Mehr Zuverlässigkeit durch try- und with-Anweisungen .....	259
9.2.1	try...finally .....	260
9.2.2	with-Anweisungen .....	261
9.3	Objekte speichern mit pickle .....	262
9.3.1	Funktionen zum Speichern und Laden. ....	262
9.4	Die Pseudofiles sys.stdin und sys.stdout .....	264
9.5	Ausgabe von Werten mit der print()-Funktion .....	265
9.5.1	Anwendung: Ausgabe von Tabellen .....	266
9.6	Kommandozeilen-Argumente (Optionen) .....	267
9.7	Aufgaben .....	270
9.8	Lösungen .....	272
<b>10</b>	<b>Definition eigener Klassen</b> .....	<b>277</b>
10.1	Klassen und Objekte .....	277
10.2	Definition von Klassen .....	279
10.3	Objekte (Instanzen) .....	281

10.4	Zugriff auf Attribute – Sichtbarkeit .....	284
10.4.1	Öffentliche Attribute .....	284
10.4.2	Private Attribute .....	285
10.4.3	Properties .....	287
10.4.4	Dynamische Erzeugung von Attributen .....	289
10.5	Methoden .....	289
10.5.1	Polymorphismus – Überladen von Operatoren .....	290
10.5.2	Vertiefung: Objekte ausführbar machen – die Methode <code>__call__()</code> .....	294
10.5.3	Statische Methoden .....	294
10.6	Abstraktion, Verkapselung und Geheimnisprinzip .....	296
10.7	Vererbung .....	297
10.7.1	Spezialisierungen .....	297
10.7.2	Beispiel: Die Klasse Konto – eine Spezialisierung der Klasse Geld .....	298
10.7.3	Vertiefung: Standardklassen als Basisklassen .....	301
10.8	Hinweise zum Programmierstil .....	303
10.8.1	Bezeichner .....	303
10.8.2	Sichtbarkeit .....	303
10.8.3	Dokumentation von Klassen .....	304
10.9	Typische Fehler .....	305
10.10	Aufgaben .....	306
10.11	Lösungen .....	310
<b>11</b>	<b>Klassenbibliotheken in Modulen speichern .....</b>	<b>315</b>
11.1	Testen einer Klasse in einem lauffähigen Stand-alone-Skript .....	315
11.2	Module speichern und importieren .....	317
11.3	Den Zugang zu einem Modul sicherstellen .....	319
11.4	Programmierstil: Verwendung und Dokumentation von Modulen .....	321
<b>12</b>	<b>Objektorientiertes Modellieren .....</b>	<b>323</b>
12.1	Phasen einer objektorientierten Software-Entwicklung .....	323
12.2	Fallstudie: Modell eines Wörterbuchs .....	324
12.2.1	OOA: Entwicklung einer Klassenstruktur .....	324
12.2.2	OOD: Entwurf einer Klassenstruktur für eine Implementierung in Python .....	328
12.2.3	OOP: Implementierung der Klassenstruktur .....	330
12.3	Assoziationen zwischen Klassen .....	334
12.3.1	Reflexive Assoziationen .....	334
12.3.2	Aggregation .....	336
12.4	Beispiel: Management eines Musicals .....	337
12.4.1	OOA .....	337
12.4.2	OOD .....	339
12.4.3	OOP .....	339
12.5	Aufgaben .....	349
12.6	Lösungen .....	350

<b>13</b>	<b>Textverarbeitung</b> .....	<b>355</b>
13.1	Standardmethoden zur Verarbeitung von Zeichenketten .....	355
13.1.1	Formatieren .....	356
13.1.2	Schreibweise .....	356
13.1.3	Tests .....	357
13.1.4	Entfernen und Aufspalten .....	358
13.1.5	Suchen und Ersetzen .....	359
13.2	Codierung und Decodierung .....	359
13.2.1	Platonische Zeichen und Unicode .....	359
13.2.2	Vertiefung: Zeichenketten durch Bytefolgen darstellen .....	361
13.3	Automatische Textproduktion .....	363
13.3.1	Texte mit variablen Teilen – Anwendung der String-Methode format() .....	363
13.3.2	Vertiefung: Eine Tabelle erstellen .....	366
13.3.3	Mahnbriefe .....	367
13.3.4	Textuelle Repräsentation eines Objektes .....	368
13.3.5	F-Strings .....	370
13.4	Analyse von Texten .....	371
13.4.1	Chat Bots .....	371
13.4.2	Textanalyse mit einfachen Vorkommenstests .....	372
13.5	Reguläre Ausdrücke .....	374
13.5.1	Aufbau eines regulären Ausdrucks .....	375
13.5.2	Objekte für reguläre Ausdrücke (RE-Objekte) .....	378
13.5.3	Analyse von Strings mit match() und search() .....	379
13.5.4	Textpassagen extrahieren mit findall() .....	379
13.5.5	Zeichenketten zerlegen mit split() .....	381
13.5.6	Teilstrings ersetzen mit sub() .....	382
13.5.7	Match-Objekte .....	382
13.6	Den Computer zum Sprechen bringen – Sprachsynthese .....	385
13.6.1	Buchstabieren .....	387
13.6.2	Den Klang der Stimme verändern .....	388
13.7	Aufgaben .....	391
13.8	Lösungen .....	394
<b>14</b>	<b>Systemfunktionen</b> .....	<b>403</b>
14.1	Das Modul sys – die Schnittstelle zum Laufzeitsystem .....	403
14.1.1	Informationen über die aktuelle Systemumgebung .....	404
14.1.2	Standardeingabe und -ausgabe .....	405
14.1.3	Die Objektverwaltung beobachten mit getrefcount() .....	406
14.1.4	Ausführung eines Skripts beenden .....	407
14.2	Das Modul os – die Schnittstelle zum Betriebssystem .....	407
14.2.1	Dateien und Verzeichnisse suchen .....	408
14.2.2	Hintergrund: Zugriffsrechte abfragen und ändern (Windows und Unix) .....	409
14.2.3	Dateien und Verzeichnisse anlegen und modifizieren .....	411

14.2.4	Merkmale von Dateien und Verzeichnissen abfragen . . . . .	412
14.2.5	Pfade verarbeiten . . . . .	413
14.2.6	Hintergrund: Umgebungsvariablen . . . . .	415
14.2.7	Systematisches Durchlaufen eines Verzeichnisbaumes . . . . .	416
14.3	Datum und Zeit . . . . .	418
14.3.1	Funktionen des Moduls time . . . . .	418
14.3.2	Sekundenformat . . . . .	419
14.3.3	Zeit-Tupel . . . . .	420
14.3.4	Zeitstrings . . . . .	421
14.3.5	Einen Prozess unterbrechen mit sleep() . . . . .	422
14.4	Zeitberechnungen mit dem Modul datetime . . . . .	422
14.4.1	Die Klasse datetime . . . . .	422
14.4.2	Die Zeitzone . . . . .	424
14.4.3	Die Klasse timedelta . . . . .	425
14.5	Aufgaben . . . . .	425
14.6	Lösungen . . . . .	426
<b>15</b>	<b>Grafische Benutzungsoberflächen mit tkinter . . . . .</b>	<b>431</b>
15.1	Ein einführendes Beispiel . . . . .	432
15.2	Einfache Widgets . . . . .	435
15.3	Die Master-Slave-Hierarchie . . . . .	436
15.4	Optionen der Widgets . . . . .	437
15.4.1	Optionen bei der Instanziierung setzen . . . . .	437
15.4.2	Widget-Optionen nachträglich konfigurieren . . . . .	438
15.4.3	Fonts . . . . .	439
15.4.4	Farben . . . . .	440
15.4.5	Rahmen . . . . .	440
15.4.6	Die Größe eines Widgets . . . . .	441
15.4.7	Leerraum um Text . . . . .	443
15.5	Gemeinsame Methoden der Widgets . . . . .	444
15.6	Die Klasse Tk . . . . .	444
15.7	Die Klasse Button . . . . .	445
15.8	Die Klasse Label . . . . .	445
15.8.1	Dynamische Konfiguration der Beschriftung . . . . .	446
15.8.2	Verwendung von Kontrollvariablen . . . . .	447
15.9	Die Klasse Entry . . . . .	449
15.10	Die Klasse Radiobutton . . . . .	451
15.11	Die Klasse Checkbutton . . . . .	453
15.12	Die Klasse Scale . . . . .	455
15.13	Die Klasse Frame . . . . .	457
15.14	Aufgaben . . . . .	457
15.15	Lösungen . . . . .	458
<b>16</b>	<b>Layout . . . . .</b>	<b>463</b>
16.1	Der Packer . . . . .	463



16.2	Layout-Fehler .....	465
16.3	Raster-Layout .....	466
16.4	Vorgehensweise bei der GUI-Entwicklung .....	470
16.4.1	Die Benutzungsoberfläche gestalten .....	473
16.4.2	Funktionalität hinzufügen .....	476
16.5	Aufgaben .....	477
16.6	Lösungen .....	480
<b>17</b>	<b>Grafik .....</b>	<b>491</b>
17.1	Die tkinter-Klasse Canvas .....	491
17.1.1	Generierung grafischer Elemente – ID, Positionierung und Display-Liste. ....	492
17.1.2	Grafische Elemente gestalten .....	494
17.1.3	Visualisieren mit Kreisdiagrammen .....	496
17.2	Die Klasse PhotoImage .....	499
17.2.1	Eine Pixelgrafik erzeugen .....	500
17.2.2	Fotos analysieren und verändern .....	502
17.3	Bilder in eine Benutzungsoberfläche einbinden .....	505
17.3.1	Icons auf Schaltflächen .....	505
17.3.2	Hintergrundbilder .....	506
17.3.3	Hintergrund: Das PPM-Format .....	508
17.4	Die Python Imaging Library (PIL) .....	509
17.4.1	Installation eines Moduls mit pip .....	509
17.4.2	Mit PIL beliebige Bilddateien einbinden .....	510
17.4.3	Steganografie – Informationen in Bildern verstecken .....	511
17.5	Aufgaben .....	513
17.6	Lösungen .....	514
<b>18</b>	<b>Event-Verarbeitung .....</b>	<b>519</b>
18.1	Einführendes Beispiel .....	520
18.2	Event-Sequenzen .....	522
18.2.1	Event-Typen .....	522
18.2.2	Qualifizierer für Maus- und Tastatur-Events .....	522
18.2.3	Modifizierer .....	524
18.3	Beispiel: Tastaturereignisse verarbeiten .....	524
18.4	Programmierung eines Eventhandlers .....	526
18.4.1	Beispiel für eine Event-Auswertung .....	527
18.5	Bindemethoden .....	528
18.6	Aufgaben .....	528
18.7	Lösungen .....	530
<b>19</b>	<b>Komplexe Benutzungsoberflächen .....</b>	<b>537</b>
19.1	Text-Widgets .....	537
19.1.1	Methoden der Text-Widgets .....	538
19.2	Rollbalken (Scrollbars) .....	540

19.3	Menüs. ....	542
19.3.1	Die Klasse Menu. ....	542
19.3.2	Methoden der Klasse Menu. ....	543
19.4	Texteditor mit Menüleiste und Pulldown-Menü. ....	544
19.5	Dialogboxen. ....	546
19.6	Applikationen mit mehreren Fenstern. ....	550
19.7	Aufgaben. ....	553
19.8	Lösungen. ....	554
<b>20</b>	<b>Threads. ....</b>	<b>559</b>
20.1	Funktionen in einem Thread ausführen. ....	560
20.2	Thread-Objekte erzeugen – die Klasse Thread. ....	562
20.3	Aufgaben. ....	565
20.4	Lösungen. ....	566
<b>21</b>	<b>Fehler finden und vermeiden. ....</b>	<b>571</b>
21.1	Testen von Bedingungen. ....	571
21.1.1	Ausnahmen (Exceptions). ....	571
21.1.2	Testen von Vor- und Nachbedingungen mit assert. ....	572
21.1.3	Vertiefung: Programmabstürze ohne Fehlermeldung. ....	575
21.2	Debugging-Modus und optimierter Modus. ....	577
21.3	Ausnahmen gezielt auslösen. ....	578
21.4	Selbstdokumentation. ....	579
21.5	Dokumentation eines Programmlaufs mit Log-Dateien. ....	581
21.5.1	Grundfunktionen. ....	581
21.5.2	Beispiel: Logging in der GUI-Programmierung. ....	582
21.6	Vertiefung: Professionelles Arbeiten mit Logging. ....	583
21.6.1	Logging-Levels. ....	583
21.6.2	Logger-Objekte. ....	588
21.6.3	Das Format der Logging-Meldungen konfigurieren. ....	588
21.7	Debugging. ....	590
21.8	Aufgabe. ....	591
21.9	Lösung. ....	592
<b>22</b>	<b>Dynamische Webseiten – CGI und WSGI. ....</b>	<b>593</b>
22.1	Wie funktionieren dynamische Webseiten? ....	593
22.2	Wie spät ist es? Aufbau eines CGI-Skripts. ....	595
22.2.1	Ein einfacher HTTP-Server. ....	599
22.3	Kommunikation über interaktive Webseiten. ....	599
22.3.1	Aufbau eines HTML-Formulars. ....	600
22.3.2	Eingabekomponenten in einem HTML-Formular. ....	602
22.4	Verarbeitung von Eingabedaten mit FieldStorage. ....	604
22.5	Sonderzeichen handhaben. ....	606
22.6	CGI-Skripte debuggen. ....	607
22.7	Der Apache-Webserver. ....	608
22.7.1	Den Apache-Server installieren. ....	608

22.7.2	CGI-Skripte auf dem Apache-Server	610
22.8	Dynamische Webseiten mit WSGI	610
22.8.1	Einfacher geht's nicht: Ein Stand-alone-WSGI-Webserver mit wsgiref	610
22.9	mod_wsgi	611
22.9.1	Installation	611
22.9.2	Vorbereitung	612
22.9.3	Den Apache-Server konfigurieren	612
22.9.4	Ein WSGI-Skript für den Apache-Server	614
22.9.5	Tipps zum Debuggen	615
22.9.6	Zugriff von einem entfernten Rechner im WLAN	616
22.10	Verarbeitung von Eingabedaten aus Formularen	616
22.11	Objektorientierte WSGI-Skripte – Beispiel: ein Chatroom	619
22.11.1	Die HTML-Seiten	621
22.11.2	Die Klassen für den Chatroom	622
22.11.3	Skript (Teil 2):	623
22.12	WSGI-Skripte mit Cookies	626
22.12.1	Besuche zählen	627
22.13	Aufgabe	629
22.14	Lösung	630
<b>23</b>	<b>Internet-Programmierung</b>	<b>635</b>
23.1	Was ist ein Protokoll?	635
23.2	Übertragung von Dateien mit FTP	636
23.2.1	Das Modul ftplib	636
23.2.2	Navigieren und Downloaden	637
23.2.3	Ein Suchroboter für FTP-Server	639
23.3	Zugriff auf Webseiten mit HTTP und HTTPS	644
23.3.1	Automatische Auswertung von Webseiten	645
23.4	Zugriff auf Ressourcen im Internet über deren URL	647
23.4.1	Webseite herunterladen und verarbeiten	647
23.4.2	Projekt: Wie warm wird es heute?	648
23.4.3	Datei herunterladen und speichern	649
23.4.4	Projekt: Filme herunterladen	649
23.5	E-Mails senden mit SMTP	651
23.6	Aufgaben	653
23.7	Lösungen	655
<b>24</b>	<b>Datenbanken</b>	<b>663</b>
24.1	Was ist ein Datenbanksystem?	663
24.2	Entity-Relationship-Diagramme (ER-Diagramme)	664
24.3	Relationale Datenbanken	665
24.4	Darstellung von Relationen als Listen oder Dictionaries	666
24.5	Das Modul sqlite3	667
24.5.1	Eine Tabelle anlegen	667

24.5.2	Anfragen an eine Datenbank . . . . .	669
24.5.3	SQL-Anweisungen mit variablen Teilen . . . . .	670
24.5.4	SQL-Injections . . . . .	671
24.6	Online-Redaktionssystem mit Datenbankanbindung . . . . .	672
24.6.1	Objektorientierte Analyse (OOA). . . . .	674
24.6.2	Objektorientierter Entwurf des Systems (OOD). . . . .	674
24.6.3	Hintergrund: Authentifizieren mit MD5-Fingerprints . . . . .	676
24.6.4	Implementierung des Redaktionssystems mit Python (OOP) . . . . .	677
24.7	Aufgaben . . . . .	687
24.8	Lösungen . . . . .	688
<b>25</b>	<b>Testen und Tuning . . . . .</b>	<b>691</b>
25.1	Automatisiertes Testen . . . . .	691
25.2	Testen mit Docstrings – das Modul doctest . . . . .	691
25.3	Praxisbeispiel: Suche nach dem Wort des Jahres . . . . .	694
25.4	Klassen testen mit doctest. . . . .	701
25.4.1	Wie testet man eine Klasse?. . . . .	701
25.4.2	Normalisierte Whitespaces – doctest-Direktiven . . . . .	702
25.4.3	Ellipsen verwenden. . . . .	702
25.4.4	Dictionaries testen . . . . .	703
25.5	Gestaltung von Testreihen mit unittest. . . . .	703
25.5.1	Einführendes Beispiel mit einem Testfall . . . . .	704
25.5.2	Klassen des Moduls unittest . . . . .	705
25.5.3	Weiterführendes Beispiel . . . . .	708
25.6	Tuning . . . . .	711
25.6.1	Performance-Analyse mit dem Profiler . . . . .	711
25.6.2	Praxisbeispiel: Auswertung astronomischer Fotografien . . . . .	713
25.6.3	Performance-Analyse und Tuning . . . . .	719
25.7	Aufgaben . . . . .	720
25.8	Lösungen . . . . .	722
<b>26</b>	<b>XML und JSON . . . . .</b>	<b>729</b>
26.1	Was ist XML? . . . . .	729
26.2	XML-Dokumente. . . . .	730
26.3	Ein XML-Dokument als Baum . . . . .	732
26.4	DOM. . . . .	733
26.5	Das Modul xml.dom.minidom. . . . .	736
26.5.1	XML-Dokumente und DOM-Objekte . . . . .	736
26.5.2	Die Basisklasse Node . . . . .	738
26.5.3	Die Klassen Document, Element und Text. . . . .	740
26.6	Attribute von XML-Elementen . . . . .	742
26.7	Anwendungsbeispiel 1: Eine XML-basierte Klasse . . . . .	742
26.8	Anwendungsbeispiel 2: Datenkommunikation mit XML . . . . .	745
26.8.1	Überblick. . . . .	746
26.8.2	Das Client-Programm. . . . .	747

26.8.3	Das Server-Programm . . . . .	750
26.9	JSON . . . . .	754
26.9.1	JSON-Texte decodieren . . . . .	755
26.9.2	Decodierungsfehler . . . . .	756
26.9.3	Ein Dictionary als JSON-Objekt speichern: Kompakt oder gut lesbar? . . . . .	756
26.9.4	Projekt: Verarbeitung von Wetterdaten . . . . .	759
26.10	Aufgaben . . . . .	762
26.11	Lösungen . . . . .	763
<b>27</b>	<b>Modellieren mit Kellern, Schlangen und Graphen . . . . .</b>	<b>765</b>
27.1	Stack (Keller, Stapel) . . . . .	765
27.2	Queue (Schlange) . . . . .	768
27.3	Graphen . . . . .	769
27.4	Aufgaben . . . . .	779
27.5	Lösungen . . . . .	781
<b>28</b>	<b>Benutzungsoberflächen mit Qt . . . . .</b>	<b>785</b>
28.1	Was bietet PyQt5? . . . . .	785
28.1.1	PyQt5 erkunden . . . . .	786
28.2	Wie arbeitet PyQt? Applikation und Fenster . . . . .	786
28.3	Eine objektorientierte Anwendung mit PyQt5 . . . . .	787
28.4	Ein Webbrowser . . . . .	788
28.5	Interaktive Widgets . . . . .	792
28.6	Label – Ausgabe von Text und Bild . . . . .	793
28.7	Signale . . . . .	794
28.8	Checkboxen und Radiobuttons . . . . .	795
28.9	Auswahlliste (ComboBox) . . . . .	798
28.10	Gemeinsame Operationen der Widgets . . . . .	800
28.11	Spezielle Methoden eines Fensters . . . . .	801
28.12	Events . . . . .	803
28.13	Fonts . . . . .	804
28.14	Stylesheets . . . . .	806
28.15	Icons . . . . .	809
28.16	Messageboxen . . . . .	809
28.17	Timer . . . . .	810
28.18	Das Qt-Layout unter der Lupe . . . . .	812
28.18.1	Absolute Positionierung und Größe . . . . .	812
28.18.2	Raster-Layout . . . . .	814
28.18.3	Form-Layout . . . . .	815
28.19	Browser für jeden Zweck . . . . .	817
28.19.1	Die Klasse QWebEngineView . . . . .	817
28.20	Ein Webbrowser mit Filter . . . . .	818
28.21	Surfen mit Geschichte – der Verlauf einer Sitzung . . . . .	820
28.22	Aufgaben . . . . .	822
28.23	Lösungen . . . . .	823

<b>29</b>	<b>Multimediaanwendungen mit Qt</b> .....	<b>827</b>
29.1	Kalender und Textfeld – ein digitales Tagebuch .....	827
	29.1.1 Programmierung .....	828
29.2	Kamerabilder .....	833
29.3	Dialoge .....	835
	29.3.1 Projekt Ansichtskarte .....	837
29.4	Videoplayer .....	841
	29.4.1 Ein einfacher Videoplayer .....	841
	29.4.2 Videoplayer mit Playlist .....	845
	29.4.3 Regeln zur Änderung der Größe (Size Policy) .....	848
	29.4.4 Das Dashboard bei Mausbewegungen einblenden .....	849
29.5	Aufgaben .....	852
29.6	Lösungen .....	856
<b>30</b>	<b>Rechnen mit NumPy</b> .....	<b>865</b>
30.1	NumPy installieren .....	865
30.2	Arrays erzeugen .....	865
	30.2.1 Arrays .....	866
	30.2.2 Matrizen und Vektoren .....	868
	30.2.3 Zahlenfolgen .....	868
	30.2.4 Zufallsarrays .....	869
	30.2.5 Spezielle Arrays .....	870
30.3	Indizieren .....	871
30.4	Slicing .....	872
30.5	Arrays verändern .....	873
30.6	Arithmetische Operationen .....	875
30.7	Funktionen, die elementweise ausgeführt werden .....	876
30.8	Einfache Visualisierung .....	877
30.9	Matrizenmultiplikation mit dot() .....	878
30.10	Array-Funktionen und Achsen .....	879
30.11	Projekt: Diffusion .....	881
30.12	Vergleiche .....	884
30.13	Projekt: Wolken am Himmel .....	884
30.14	Projekt: Wie versteckt man ein Buch in einem Bild? .....	887
30.15	Datenanalyse mit Histogrammen .....	890
30.16	Wie funktioniert ein Medianfilter? .....	893
30.17	Rechnen mit SciPy .....	896
	30.17.1 Lineare Gleichungssysteme lösen .....	896
	30.17.2 Integration .....	898
30.18	Aufgaben .....	899
30.19	Lösungen .....	902
<b>31</b>	<b>Messdaten verarbeiten</b> .....	<b>907</b>
31.1	Messwerte in einem Diagramm darstellen – Matplotlib und tkinter .....	907
	31.1.1 Basisprojekt .....	907

31.1.2	Erweiterung: Den letzten Wert löschen . . . . .	911
31.1.3	Das Aussehen eines Diagramms gestalten . . . . .	913
31.2	Messwerte aus einem Multimeter lesen und darstellen . . . . .	916
31.2.1	Vorbereitung . . . . .	916
31.2.2	Werte auslesen . . . . .	917
31.2.3	Welche Ziffern zeigt das Display des Multimeters? . . . . .	920
31.3	Anzeige der Temperatur . . . . .	924
31.4	Messreihen aufzeichnen . . . . .	926
31.5	Aufgabe. . . . .	929
31.6	Lösung . . . . .	929
<b>32</b>	<b>Parallele Datenverarbeitung . . . . .</b>	<b>933</b>
32.1	Was sind parallele Programme? . . . . .	933
32.2	Prozesse starten und abrechnen . . . . .	934
32.3	Funktionen in eigenen Prozessen starten . . . . .	935
32.4	Prozesse zusammenführen – join() . . . . .	937
32.5	Wie können Prozesse Objekte austauschen? . . . . .	938
32.5.1	Objekte als Argumente übergeben . . . . .	938
32.5.2	Objekte über eine Pipe senden und empfangen. . . . .	938
32.5.3	Objekte über eine Queue austauschen . . . . .	939
32.6	Daten im Pool bearbeiten . . . . .	940
32.6.1	Mit dem Pool geht's schneller – ein Zeitexperiment . . . . .	941
32.6.2	Forschen mit Big Data aus dem Internet . . . . .	942
32.7	Synchronisation . . . . .	945
32.8	Produzenten und Konsumenten . . . . .	948
32.8.1	Sprücheklopfer . . . . .	949
32.9	Aufgaben . . . . .	951
32.10	Lösungen . . . . .	952
<b>33</b>	<b>Django. . . . .</b>	<b>955</b>
33.1	Django aus der Vogelperspektive . . . . .	955
33.2	Start eines Projekts . . . . .	956
33.2.1	Den Server starten. . . . .	958
33.2.2	Startseite und View einrichten. . . . .	959
33.3	Datenbankanbindung . . . . .	961
33.4	Modelle erstellen . . . . .	962
33.5	Modelle aktivieren . . . . .	963
33.6	In der Python-Shell die Datenbank bearbeiten . . . . .	967
33.6.1	Objekte durch Aufruf der Klasse erzeugen . . . . .	967
33.6.2	Auf Attribute eines Objektes zugreifen . . . . .	968
33.6.3	Objekte finden . . . . .	969
33.6.4	Objekte erzeugen und Beziehungen herstellen . . . . .	970
33.6.5	Den Beziehungsmanager nutzen. . . . .	970
33.6.6	Objekte löschen. . . . .	971
33.7	Django-Modelle unter der Lupe . . . . .	971

33.8	Der Manager unter der Lupe – Objekte erzeugen und suchen . . . . .	973
33.9	Administration . . . . .	976
	33.9.1 Eine Applikation der Website-Verwaltung zugänglich machen . . . . .	978
33.10	Views einrichten – die Grundstruktur. . . . .	982
	33.10.1 Was sind Views? . . . . .	982
	33.10.2 Funktionen für Views . . . . .	983
	33.10.3 URL-Patterns. . . . .	984
33.11	View-Funktionen erweitern . . . . .	985
	33.11.1 Startseite . . . . .	985
	33.11.2 Auflistung der Ideen zu einer Frage – question_index . . . . .	988
	33.11.3 Die Templates verbessern: Namen statt expliziter URLs. . . . .	990
33.12	Interaktive Webseiten – Views mit Formularen . . . . .	991
	33.12.1 Eingabe einer neuen Frage . . . . .	991
	33.12.2 Eingabe einer neuen Idee. . . . .	996
	33.12.3 View-Funktion für das Speichern einer neuen Idee . . . . .	997
	33.12.4 Fertig! . . . . .	998
33.13	Die nächsten Schritte . . . . .	998
33.14	Aufgabe Suche nach Ideen . . . . .	999
33.15	Lösung . . . . .	1000
<b>A</b>	<b>Anhang. . . . .</b>	<b>1003</b>
A.1	Zeichencodierung . . . . .	1003
	A.1.1 Codierung von Sonderzeichen in HTML . . . . .	1003
A.2	Quellen im WWW. . . . .	1003
A.3	Standardfunktionen und Standardklassen . . . . .	1004
A.4	Mathematische Funktionen . . . . .	1006
	A.4.1 Das Modul math . . . . .	1006
	A.4.2 Das Modul random. . . . .	1007
A.5	EBNF-Grammatik . . . . .	1008
<b>B</b>	<b>Glossar . . . . .</b>	<b>1013</b>
<b>C</b>	<b>Download der Programmbeispiele. . . . .</b>	<b>1025</b>
<b>D</b>	<b>Ein Python-Modul veröffentlichen: PyPI. . . . .</b>	<b>1027</b>
D.1	Bei PyPI und TestPyPI registrieren . . . . .	1028
D.2	Ein Paket für die Veröffentlichung vorbereiten . . . . .	1029
	D.2.1 Die Programmdatei setup.py . . . . .	1029
	D.2.2 Die Lizenz . . . . .	1030
	D.2.3 Die Datei README.txt . . . . .	1031
	D.2.4 Die Datei __init__.py . . . . .	1032
D.3	Das Paket auf PyPI veröffentlichen . . . . .	1032
	D.3.1 Das Paket aktualisieren . . . . .	1033
	<b>Stichwortverzeichnis . . . . .</b>	<b>1035</b>