

Inhaltsverzeichnis

	Einleitung	23
	Warum Python?	23
	Python 3	23
	An wen wendet sich dieses Buch?	23
	Inhalt und Aufbau	24
	Hinweise zur Typographie	25
	Programmbeispiele	26
I	Grundlagen	27
I.1	Was ist Programmieren?	27
I.2	Hardware und Software	28
I.3	Programm als Algorithmus	29
I.4	Syntax und Semantik	30
I.5	Interpreter und Compiler	30
I.6	Programmierparadigmen	32
I.7	Objektorientierte Programmierung	33
I.7.1	Strukturelle Zerlegung	33
I.7.2	Die Welt als System von Objekten	34
I.7.3	Objekte besitzen Attribute und beherrschen Methoden	35
I.7.4	Objekte sind Instanzen von Klassen	36
I.8	Hintergrund: Geschichte der objektorientierten Programmierung	36
I.9	Aufgaben	37
I.10	Lösungen	38
2	Der Einstieg – Python im interaktiven Modus	39
2.1	Python installieren	39
2.2	Python im interaktiven Modus	42
2.2.1	Start des Python-Interpreters in einem Konsolenfenster ...	42
2.2.2	Die Python-Shell von IDLE	42
2.2.3	Die ersten Python-Befehle ausprobieren	43
2.2.4	Hotkeys	43
2.3	Objekte	44

2.4	Namen	46
2.5	Hintergrund: Syntax-Regeln für Bezeichner	47
2.6	Schlüsselwörter	48
2.7	Anweisungen	48
2.7.1	Ausdrucksanweisungen	48
2.7.2	Import-Anweisungen	54
2.7.3	Zuweisungen	55
2.7.4	Erweiterte Zuweisungen	58
2.7.5	Hintergrund: Dynamische Typisierung	59
2.8	Aufgaben	60
2.9	Lösungen	61
3	Python-Skripte	65
3.1	Skripte editieren und ausführen mit IDLE	65
3.2	Ausführen eines Python-Skripts	66
3.3	Kommentare	69
3.4	Die Zeilenstruktur von Python-Programmen	69
3.5	Das EVA-Prinzip	73
3.6	Phasen der Programmentwicklung	74
3.7	Guter Programmierstil	75
3.8	Hintergrund: Die Kunst des Fehlerfindens	78
3.9	Aufgaben	80
3.10	Lösungen	81
4	Standard-Datentypen	83
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	106
4.11	Mengen	107
4.12	Dictionaries	108
4.13	Typumwandlungen	108
4.13.1	int()	109
4.13.2	float()	110
4.13.3	complex()	111
4.13.4	bool()	111
4.13.5	str()	111
4.13.6	dict(), list() und tuple()	112
4.14	Aufgaben	112
4.15	Lösungen	115
5	Kontrollstrukturen	119
5.1	Einfache Bedingungen	119
5.1.1	Vergleiche	119
5.1.2	Zugehörigkeit zu einer Menge (in, not in)	123
5.1.3	Beliebige Ausdrücke als Bedingungen	123
5.2	Zusammengesetzte Bedingungen – logische Operatoren	124
5.2.1	Negation (not)	124
5.2.2	Konjunktion (and)	125
5.2.3	Disjunktion (or)	126
5.2.4	Formalisierung von Bedingungen	127
5.2.5	Hinweis zum Programmierstil	128
5.3	Programmverzweigungen (bedingte Anweisungen)	128
5.3.1	Einseitige Verzweigung (if)	129
5.3.2	Zweiseitige Verzweigung (if-else)	129
5.3.3	Mehrfache Fallunterscheidung (elif)	130
5.3.4	Bedingte Ausdrücke	132
5.4	Bedingte Wiederholung (while)	132
5.4.1	Endlosschleifen	133
5.5	Iteration über eine Kollektion (for)	135
5.5.1	Zählschleifen – Verwendung von range()	136
5.5.2	Verschachtelte Iterationen	137
5.5.3	Vertiefung: Iterative Berechnung rekursiver Folgen	139

5.6	Abbruch einer Schleife mit break	139
5.6.1	Abbruch eines Schleifendurchlaufs mit continue	140
5.7	Abfangen von Ausnahmen mit try	141
5.7.1	try...except	142
5.8	Aufgaben	144
5.9	Lösungen	148
6	Funktionen	153
6.1	Aufruf von Funktionen	153
6.2	Definition von Funktionen	156
6.3	Schrittweise Verfeinerung	158
6.4	Ausführung von Funktionen	161
6.4.1	Globale und lokale Namen	161
6.4.2	Seiteneffekte – die global-Anweisung	165
6.4.3	Parameterübergabe	166
6.5	Voreingestellte Parameterwerte	168
6.5.1	Schlüsselwort-Argumente	170
6.6	Funktionen mit beliebiger Anzahl von Parametern	172
6.7	Lokale Funktionen	173
6.8	Rekursive Funktionen	174
6.9	Experimente zur Rekursion mit der Turtle-Grafik	176
6.9.1	Turtle-Befehle im interaktiven Modus	176
6.9.2	Eine rekursive Spirale	177
6.9.3	Baumstrukturen	179
6.9.4	Künstlicher Blumenkohl – selbstähnliche Bilder	180
6.10	Rekursive Zahlenfunktionen	182
6.11	Hintergrund: Wie werden rekursive Funktionen ausgeführt?	183
6.11.1	Execution Frames	183
6.11.2	Rekursionstiefe	184
6.12	Funktionen als Objekte	186
6.12.1	Hintergrund: Typen sind keine Funktionen	187
6.13	Lambda-Formen	187
6.14	Hinweise zum Programmierstil	188
6.14.1	Allgemeines	188
6.14.2	Funktionsnamen	188
6.14.3	Kommentierte Parameter	189
6.14.4	Docstrings	189
6.15	Aufgaben	190
6.16	Lösungen	193

7	Sequenzen, Mengen und Generatoren	197
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	205
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
8	Dictionaries	237
8.1	Operationen für Dictionaries	237
8.2	Wie erstellt man ein Dictionary?	238
8.2.1	Definition mit einem Dictionary-Display	238
8.2.2	Schrittweiser Aufbau eines Dictionarys	240
8.2.3	Ein Dictionary aus anderen Dictionaries zusammensetzen – update()	241
8.3	Zugriff auf Daten in einem Dictionary	241
8.3.1	Vergebliche Zugriffsversuche	241
8.4	Praxisbeispiel: Vokabeltrainer	242

8.5	Typische Fehler	244
8.6	Aufgaben	244
8.7	Lösungen	247
9	Ein- und Ausgabe	251
9.1	Files	251
9.1.1	Die Rolle der Files bei E/A-Operationen	251
9.1.2	Was ist ein File?	252
9.1.3	Ein File-Objekt erzeugen	253
9.1.4	Speichern einer Zeichenkette	254
9.1.5	Laden einer Zeichenkette aus einer Datei	255
9.1.6	Absolute und relative Pfade	255
9.1.7	Zwischenspeichern, ohne zu schließen	258
9.1.8	Zugriff auf Files (lesen und schreiben)	258
9.1.9	Speichern beliebiger Daten auf Files	260
9.2	Mehr Zuverlässigkeit durch try- und with-Anweisungen	261
9.2.1	try...finally	262
9.2.2	With-Anweisungen	263
9.3	Objekte speichern mit pickle	264
9.3.1	Funktionen zum Speichern und Laden	265
9.4	Die Pseudofiles sys.stdin und sys.stdout	266
9.5	Ausgabe von Werten mit der print()-Funktion	267
9.5.1	Anwendung: Ausgabe von Tabellen	269
9.6	Kommandozeilen-Argumente (Optionen)	269
9.7	Übungen	272
9.8	Lösungen	275
10	Definition eigener Klassen	281
10.1	Klassen und Objekte	281
10.2	Definition von Klassen	283
10.3	Objekte (Instanzen)	285
10.4	Zugriff auf Attribute – Sichtbarkeit	288
10.4.1	Öffentliche Attribute	288
10.4.2	Private Attribute	289
10.4.3	Properties	291
10.4.4	Dynamische Erzeugung von Attributen	293
10.5	Methoden	293
10.5.1	Polymorphismus – Überladen von Operatoren	294

10.6	Statische Methoden	298
10.7	Abstraktion, Verkapselung und Geheimnisprinzip	299
10.8	Vererbung	300
10.8.1	Spezialisierungen	300
10.8.2	Beispiel: Die Klasse Konto – eine Spezialisierung der Klasse Geld	301
10.8.3	Vertiefung: Standardklassen als Basisklassen	304
10.9	Hinweise zum Programmierstil	306
10.9.1	Bezeichner	306
10.9.2	Sichtbarkeit	306
10.9.3	Dokumentation von Klassen	308
10.10	Typische Fehler	308
10.11	Aufgaben	310
10.12	Lösungen	313
11	Klassenbibliotheken in Modulen speichern	319
11.1	Testen einer Klasse in einem lauffähigen Stand-alone-Skript	319
11.2	Module speichern und importieren	321
11.3	Den Zugang zu einem Modul sicherstellen	323
11.4	Programmierstil: Verwendung und Dokumentation von Modulen	325
12	Objektorientiertes Modellieren	327
12.1	Phasen einer objektorientierten Software-Entwicklung	327
12.2	Fallstudie: Modell eines Wörterbuchs	328
12.2.1	OOA: Entwicklung einer Klassenstruktur	328
12.2.2	OOD: Entwurf einer Klassenstruktur für eine Implementierung in Python	332
12.2.3	OOP: Implementierung der Klassenstruktur	334
12.3	Assoziationen zwischen Klassen	338
12.3.1	Reflexive Assoziationen	338
12.3.2	Aggregation	340
12.4	Beispiel: Management eines Musicals	341
12.4.1	OOA	341
12.4.2	OOD	343
12.4.3	OOP	343
12.5	Aufgaben	353
12.6	Lösungen	354

I3	Verarbeitung von Zeichenketten	359
I3.1	Standardmethoden zur Verarbeitung von Zeichenketten	359
I3.1.1	Formatieren	360
I3.1.2	Schreibweise	360
I3.1.3	Tests	361
I3.1.4	Entfernen und Aufspalten	362
I3.1.5	Suchen und Ersetzen	363
I3.2	Codierung und Decodierung	363
I3.2.1	Platonische Zeichen und Unicode	363
I3.2.2	Vertiefung: Zeichenketten durch Bytefolgen darstellen	365
I3.3	Automatische Textproduktion	367
I3.3.1	Texte mit variablen Teilen – Anwendung der String-Methode format()	367
I3.3.2	Vertiefung: Eine Tabelle erstellen	370
I3.3.3	Mahnbriefe	371
I3.3.4	Textuelle Repräsentation eines Objektes	372
I3.4	Analyse von Texten	374
I3.4.1	Chat Bots	374
I3.4.2	Textanalyse mit einfachen Vorkommenstests	375
I3.5	Reguläre Ausdrücke	377
I3.5.1	Aufbau eines regulären Ausdrucks	378
I3.5.2	Objekte für reguläre Ausdrücke (RE-Objekte)	381
I3.5.3	Analyse von Strings mit match() und search()	382
I3.5.4	Textpassagen extrahieren mit findall()	383
I3.5.5	Zeichenketten zerlegen mit split()	385
I3.5.6	Teilstrings ersetzen mit sub()	386
I3.5.7	Match-Objekte	386
I3.6	Den Computer zum Sprechen bringen – Sprachsynthese	389
I3.6.1	Buchstabieren	391
I3.6.2	Den Klang der Stimme verändern	392
I3.7	Aufgaben	395
I3.8	Lösungen	398
I4	Systemfunktionen	407
I4.1	Das Modul sys – die Schnittstelle zum Laufzeitsystem	407
I4.1.1	Informationen über die aktuelle Systemumgebung	408
I4.1.2	Standardeingabe und -ausgabe	409
I4.1.3	Die Objektverwaltung beobachten mit getrefcount()	410

14.1.4	Ausführung eines Skripts beenden	411
14.2	Das Modul os – die Schnittstelle zum Betriebssystem	411
14.2.1	Dateien und Verzeichnisse suchen	412
14.2.2	Hintergrund: Zugriffsrechte abfragen und ändern (Windows und Unix)	413
14.2.3	Dateien und Verzeichnisse anlegen und modifizieren	415
14.2.4	Merkmale von Dateien und Verzeichnissen abfragen	416
14.2.5	Pfade verarbeiten	417
14.2.6	Hintergrund: Umgebungsvariablen	419
14.2.7	Systematisches Durchlaufen eines Verzeichnisbaumes	420
14.3	Datum und Zeit	422
14.3.1	Funktionen des Moduls time	422
14.3.2	Sekundenformat	423
14.3.3	Zeit-Tupel	424
14.3.4	Zeitstrings	425
14.3.5	Einen Prozess unterbrechen mit sleep()	426
14.4	Aufgaben	426
14.5	Lösungen	428
15	Grafische Benutzungsoberflächen mit tkinter	433
15.1	Ein einführendes Beispiel	434
15.2	Einfache Widgets	437
15.3	Die Master-Slave-Hierarchie	438
15.4	Optionen der Widgets	439
15.4.1	Optionen bei der Instanziierung setzen	439
15.4.2	Widget-Optionen nachträglich konfigurieren	440
15.4.3	Fonts	441
15.4.4	Farben	442
15.4.5	Rahmen	443
15.4.6	Die Größe eines Widgets	443
15.4.7	Leerraum um Text	445
15.5	Gemeinsame Methoden der Widgets	446
15.6	Die Klasse Tk	447
15.7	Die Klasse Button	447
15.8	Die Klasse Label	448
15.8.1	Dynamische Konfiguration der Beschriftung	448
15.8.2	Verwendung von Kontrollvariablen	449

15.9	Die Klasse Entry	451
15.10	Die Klasse Radiobutton	453
15.11	Die Klasse Checkbutton	455
15.12	Die Klasse Scale	457
15.13	Die Klasse Frame	459
15.14	Aufgaben	459
15.15	Lösungen	461
16	Layout	467
16.1	Der Packer	467
16.2	Layout-Fehler	469
16.3	Raster-Layout	470
16.4	Vorgehensweise bei der GUI-Entwicklung	474
	16.4.1 Die Benutzungsoberfläche gestalten	477
	16.4.2 Funktionalität hinzufügen	480
16.5	Aufgaben	481
16.6	Lösungen	484
17	Grafik	495
17.1	Die tkinter-Klasse Canvas	495
	17.1.1 Generierung grafischer Elemente – ID, Positionierung und Display-Liste	496
	17.1.2 Grafische Elemente gestalten	498
	17.1.3 Visualisieren mit Kreisdiagrammen	500
17.2	Die Klasse PhotoImage	503
	17.2.1 Eine Pixelgrafik erzeugen	504
	17.2.2 Fotos analysieren und verändern	506
17.3	Bilder in eine Benutzungsoberfläche einbinden	509
	17.3.1 Icons auf Schaltflächen	509
	17.3.2 Hintergrundbilder	510
	17.3.3 Hintergrund: Das PPM-Format	512
17.4	Die Python Imaging Library (PIL)	513
	17.4.1 Installation eines Moduls mit pip	513
	17.4.2 Mit PIL beliebige Bilddateien einbinden	514
	17.4.3 Steganografie – Informationen in Bildern verstecken	515
17.5	Aufgaben	517
17.6	Lösungen	518

18	Event-Verarbeitung	523
18.1	Einführendes Beispiel	524
18.2	Event-Sequenzen	526
18.2.1	Event-Typen	526
18.2.2	Qualifizierer für Maus- und Tastatur-Events	526
18.2.3	Modifizierer	528
18.3	Beispiel: Tastaturereignisse verarbeiten	528
18.4	Programmierung eines Eventhandlers	530
18.4.1	Beispiel für eine Event-Auswertung	531
18.5	Bindemethoden	532
18.6	Aufgaben	532
18.7	Lösungen	535
19	Komplexe Benutzungsoberflächen	541
19.1	Text-Widgets	541
19.1.1	Methoden der Text-Widgets	542
19.2	Rollbalken (Scrollbars)	544
19.3	Menüs	546
19.3.1	Die Klasse Menu	546
19.3.2	Methoden der Klasse Menu	547
19.4	Texteditor mit Menüleiste und Pulldown-Menü	548
19.5	Dialogboxen	550
19.6	Applikationen mit mehreren Fenstern	554
19.7	Aufgaben	557
19.8	Lösungen	558
20	Threads	563
20.1	Funktionen in einem Thread ausführen	564
20.2	Thread-Objekte erzeugen – die Klasse Thread	566
20.3	Aufgaben	569
20.4	Lösungen	570
21	Fehler finden und vermeiden	575
21.1	Testen von Bedingungen	575
21.1.1	Ausnahmen (Exceptions)	575
21.1.2	Testen von Vor- und Nachbedingungen mit assert	576
21.1.3	Vertiefung: Programmabstürze ohne Fehlermeldung	579

21.2	Debugging-Modus und optimierter Modus	581
21.3	Ausnahmen gezielt auslösen	582
21.4	Selbstdokumentation	583
21.5	Dokumentation eines Programmlaufs mit Log-Dateien	585
21.5.1	Grundfunktionen	585
21.5.2	Beispiel: Logging in der GUI-Programmierung	586
21.6	Vertiefung: Professionelles Arbeiten mit Logging	587
21.6.1	Logging-Levels	587
21.6.2	Logger-Objekte	592
21.6.3	Das Format der Logging-Meldungen konfigurieren	592
21.7	Debugging	594
21.8	Aufgabe	595
21.9	Lösung	596
22	CGI-Programmierung	597
22.1	Wie funktionieren CGI-Skripte?	597
22.2	Wie spät ist es? Aufbau eines CGI-Skripts	599
22.2.1	Ein einfacher HTTP-Server	602
22.2.2	Hintergrund: CGI-Skripte auf einem Host im Internet installieren	603
22.3	Kommunikation über interaktive Webseiten	604
22.3.1	Aufbau eines HTML-Formulars	604
22.3.2	Eingabekomponenten in einem HTML-Formular	606
22.4	Verarbeitung von Eingabedaten in einem CGI-Skript	608
22.5	Sonderzeichen handhaben	610
22.6	CGI-Skripte debuggen	611
22.7	Objektorientierte CGI-Skripte – Beispiel: ein Chatroom	612
22.8	CGI-Skripte mit Cookies	618
22.9	Aufgaben	621
22.10	Lösungen	623
23	Internet-Programmierung	629
23.1	Was ist ein Protokoll?	629
23.2	Übertragung von Dateien mit FTP	630
23.2.1	Das Modul ftplib	631
23.2.2	Navigieren und Downloaden	632
23.2.3	Ein Suchroboter für FTP-Server	634

23.3	Zugriff auf Webseiten mit HTTP	638
23.3.1	Automatische Auswertung von Webseiten	640
23.4	E-Mails senden mit SMTP	642
23.5	Aufgaben	645
23.6	Lösungen	647
24	Datenbanken	653
24.1	Was ist ein Datenbanksystem?	653
24.2	Entity-Relationship-Diagramme (ER-Diagramme)	654
24.3	Relationale Datenbanken	655
24.4	Darstellung von Relationen als Listen oder Dictionaries	656
24.5	Das Modul sqlite3	657
24.5.1	Eine Tabelle anlegen	657
24.5.2	Anfragen an eine Datenbank	659
24.5.3	SQL-Anweisungen mit variablen Teilen	660
24.5.4	SQL-Injections	661
24.6	Online-Redaktionssystem mit Datenbankbindung	662
24.6.1	Objektorientierte Analyse (OOA)	664
24.6.2	Objektorientierter Entwurf des Systems (OOD)	664
24.6.3	Hintergrund: Authentifizieren mit MD5-Fingerprints	666
24.6.4	Implementierung des Redaktionssystems mit Python (OOP)	667
24.7	Aufgaben	676
24.8	Lösungen	677
25	Testen und Tuning	681
25.1	Automatisiertes Testen	681
25.2	Testen mit Docstrings – das Modul doctest	682
25.3	Praxisbeispiel: Suche nach dem Wort des Jahres	684
25.4	Klassen testen mit doctest	691
25.4.1	Wie testet man eine Klasse?	691
25.4.2	Normalisierte Whitespaces – doctest-Direktiven	692
25.4.3	Ellipsen verwenden	692
25.4.4	Dictionaries testen	693
25.5	Gestaltung von Testreihen mit unittest	693
25.5.1	Einführendes Beispiel mit einem Testfall	693
25.5.2	Klassen des Moduls unittest	695
25.5.3	Weiterführendes Beispiel	697

25.6	Tuning	701
25.6.1	Performance-Analyse mit dem Profiler	701
25.6.2	Praxisbeispiel: Auswertung astronomischer Fotografien ...	703
25.6.3	Performance-Analyse und Tuning	709
25.7	Aufgaben	710
25.8	Lösungen	712
26	XML	719
26.1	Was ist XML?	719
26.2	XML-Dokumente	720
26.3	Ein XML-Dokument als Baum	722
26.4	DOM	723
26.5	Das Modul xml.dom.minidom	726
26.5.1	XML-Dokumente und DOM-Objekte	726
26.5.2	Die Basisklasse Node	728
26.5.3	Die Klassen Document, Element und Text	730
26.6	Attribute von XML-Elementen	732
26.7	Anwendungsbeispiel 1: Eine XML-basierte Klasse	732
26.8	Anwendungsbeispiel 2: Datenkommunikation mit XML	735
26.8.1	Überblick	736
26.8.2	Das Client-Programm	737
26.8.3	Das Server-Programm	740
26.9	Aufgaben	744
26.10	Lösungen	745
27	Modellieren mit Kellern, Schlangen und Graphen	747
27.1	Stack (Keller, Stapel)	747
27.2	Queue (Schlange)	750
27.3	Graphen	751
27.4	Aufgaben	761
27.5	Lösungen	763
28	Benutzungsoberflächen mit Qt	767
28.1	Was bietet PyQt5?	767
28.1.1	PyQt5 erkunden	768
28.2	Wie arbeitet PyQt? Applikation und Fenster	768
28.3	Eine objektorientierte Anwendung mit PyQt5	769

28.4	Ein Webbrowser	770
28.5	Interaktive Widgets	773
28.6	Label – Ausgabe von Text und Bild	775
28.7	Signale	776
28.8	Checkboxen und Radiobuttons	777
28.9	Auswahlliste (ComboBox)	780
28.10	Gemeinsame Operationen der Widgets	782
28.11	Spezielle Methoden eines Fensters	783
28.12	Events	785
28.13	Fonts	786
28.14	Stylesheets	788
28.15	Icons	791
28.16	Messageboxen	791
28.17	Timer	792
28.18	Das Qt-Layout unter der Lupe	794
	28.18.1 Absolute Positionierung und Größe	795
	28.18.2 Raster-Layout	796
	28.18.3 Form-Layout	797
28.19	Browser für jeden Zweck	799
	28.19.1 Die Klasse QWebView	800
28.20	Ein Webbrowser mit Filter	800
28.21	Surfen mit Geschichte – der Verlauf einer Sitzung	802
28.22	Aufgaben	804
28.23	Lösungen	805
29	Multimediaanwendungen mit Qt	811
29.1	Kalender und Textfeld – ein digitales Tagebuch	811
	29.1.1 Programmierung	812
29.2	Kamerabilder	817
29.3	Dialoge	819
	29.3.1 Projekt Ansichtskarte	821
29.4	Videoplayer	825
	29.4.1 Ein einfacher Videoplayer	825
	29.4.2 Videoplayer mit Playlist	829
	29.4.3 Regeln zur Änderung der Größe (Size Policy)	832
	29.4.4 Das Dashboard bei Mausbewegungen einblenden	833
29.5	Aufgaben	836
29.6	Lösungen	840

30	Rechnen mit NumPy	849
30.1	NumPy installieren	849
30.2	Arrays erzeugen	849
30.2.1	Arrays	850
30.2.2	Matrizen und Vektoren	852
30.2.3	Zahlenfolgen	852
30.2.4	Zufallsarrays	853
30.2.5	Spezielle Arrays	854
30.3	Indizieren	855
30.4	Slicing	856
30.5	Arrays verändern	857
30.6	Arithmetische Operationen	859
30.7	Funktionen, die elementweise ausgeführt werden	860
30.8	Einfache Visualisierung	861
30.9	Matrizenmultiplikation mit dot()	862
30.10	Array-Funktionen und Achsen	863
30.11	Projekt: Diffusion	865
30.12	Vergleiche	868
30.13	Projekt: Wolken am Himmel	868
30.14	Projekt: Wie versteckt man ein Buch in einem Bild?	871
30.15	Datenanalyse mit Histogrammen	874
30.16	Wie funktioniert ein Medianfilter?	877
30.17	Rechnen mit SciPy	880
30.17.1	Lineare Gleichungssysteme lösen	880
30.17.2	Integration	882
30.18	Aufgaben	883
30.19	Lösungen	886
31	Messdaten verarbeiten	891
31.1	Messwerte aus einem Multimeter lesen und darstellen	891
31.1.1	Vorbereitung	891
31.1.2	Werte auslesen	892
31.1.3	Welche Ziffern zeigt das Display des Multimeters?	895
31.2	Anzeige der Temperatur	899
31.3	Messreihen aufzeichnen	901
31.4	Aufgabe	904
31.5	Lösung	904

32	Parallele Datenverarbeitung	907
32.1	Was sind parallele Programme?	907
32.2	Prozesse starten und abbrechen	908
32.3	Funktionen in eigenen Prozessen starten	909
32.4	Prozesse zusammenführen – join()	911
32.5	Wie können Prozesse Objekte austauschen?	911
32.5.1	Objekte als Argumente übergeben	911
32.5.2	Objekte über eine Pipe senden und empfangen	912
32.5.3	Objekte über eine Queue austauschen	913
32.6	Daten im Pool bearbeiten	914
32.6.1	Mit dem Pool geht's schneller – ein Zeitexperiment	915
32.6.2	Forschen mit Big Data aus dem Internet	916
32.7	Synchronisation	919
32.8	Produzenten und Konsumenten	922
32.8.1	Sprücheklopfer	922
32.9	Aufgaben	924
32.10	Lösungen	925
A	Anhang	929
A.1	Zeichencodierung	929
A.1.1	Codierung von Sonderzeichen in HTML	929
A.2	Quellen im WWW	929
A.3	Standardfunktionen und Standardklassen	930
A.4	Mathematische Funktionen	932
A.4.1	Das Modul math	932
A.4.2	Das Modul random	933
B	Glossar	935
C	Download der Programmbeispiele	947
D	Ein Python-Modul veröffentlichen: PyPI	949
D.1	Bei PyPI und TextPyPI registrieren	950
D.2	Ein Paket für die Veröffentlichung vorbereiten	951
D.2.1	Die Programmdatei setup.py	951
D.2.2	Die Lizenz	952

	D.2.3	Die Datei README.txt	953
	D.2.4	Die Datei __init__.py	954
D.3		Das Paket auf PyPI veröffentlichen	954
	D.3.1	Das Paket aktualisieren	955
		Stichwortverzeichnis	957