

<b>1</b>	<b>Einleitung und Grundlagen – Bevor es richtig losgeht</b>	<b>1</b>
1.1	Was behandeln wir in dem einleitenden Kapitel?	1
1.2	Das Ziel des Buchs.	1
1.3	Was sollten Sie bereits können?	2
1.4	Was ist Python?	2
1.4.1	Das Ziel von Python.	3
1.4.2	Was umfasst Python?	3
1.4.3	Die verschiedenen Python-Paradigma	4
1.5	Was benötigen Sie zum Arbeiten mit dem Buch?	4
1.5.1	Hardware und Betriebssystem	4
1.5.2	Die Python-Version	5
1.5.3	Python laden und installieren.	5
<b>2</b>	<b>Erste Beispiele – Der Sprung ins kalte Wasser.</b>	<b>21</b>
2.1	Was behandeln wir in diesem Kapitel?	21
2.2	Der Interaktivmodus – die Kommandozeile von Python	21
2.2.1	Das Prompt.	23
2.2.2	Der Hilfemodus in der Kommandozeile	26
2.3	Anweisungen in (echten) Quelltext auslagern	29
2.4	Von IDLE & Co bis zu Python in der Cloud	31
2.4.1	Weitere IDEs und Editoren für Python	32
2.4.2	Cloud und RIA.	33
<b>3</b>	<b>Built-in Functions – Modularisierung durch Unterprogramme</b>	<b>41</b>
3.1	Was behandeln wir in diesem Kapitel?	41
3.2	Was sind Funktionen im Allgemeinen?	41
3.3	Built-in-Funktionen	42
3.3.1	Hilfe zu Built-in Functions im Hilfemodus	42
3.3.2	Hilfe zu Built-in Functions im Editormodus	43

3.3.3	Die print()-Funktion. ....	44
3.3.4	Die input()-Funktion ....	50
3.3.5	Eine kurze Übersicht aller Built-in Functions ....	52
<b>4</b>	<b>Grundlegende Begriffe – Kommentare, SheBang und Strukturanalysen ...</b>	<b>55</b>
4.1	Was behandeln wir in diesem Kapitel? ....	55
4.2	Token und Parser ....	55
4.2.1	Zerlegen von Quelltext. ....	56
4.3	Kommentare. ....	58
4.3.1	Kommentare in Python ....	58
4.4	SheBang und eine Python-Datei direkt ausführen. ....	60
4.4.1	SheBang als besonderer Kommentar. ....	60
<b>5</b>	<b>Anweisungen – Dem Computer Befehle geben ....</b>	<b>61</b>
5.1	Was behandeln wir in diesem Kapitel? ....	61
5.2	Was sind Anweisungen? ....	61
5.2.1	Eine Frage der Reihenfolge ....	61
5.3	Anweisungsarten ....	62
5.3.1	Blockanweisung. ....	62
5.3.2	Kontrollflussanweisungen ....	63
5.3.3	Deklarationsanweisung ....	64
5.3.4	Ausdrucksanweisung ....	64
5.3.5	Die leere Anweisung <i>pass</i> ....	65
<b>6</b>	<b>Datentypen, Variablen und Literale – Die Art der Information. ....</b>	<b>67</b>
6.1	Was behandeln wir in diesem Kapitel? ....	67
6.2	Variablen ....	67
6.2.1	Variablen deklarieren ....	67
6.2.2	Variablen im Quellcode verwenden ....	69
6.3	Die Datentypen in Python ....	69
6.3.1	Lose Typisierung und Typumwandlung in Python ....	69
6.3.2	Die Python-Datentypen ....	71
6.3.3	Zahlen – <i>int</i> , <i>float</i> und <i>complex</i> ....	73
6.3.4	Zeichenketten -(str) und Zeichenlitterale ....	78
6.4	Den Datentyp bestimmen und umwandeln ....	79
6.4.1	Den Datentyp mit <i>type()</i> dynamisch bestimmen ....	79
6.4.2	Implizite und explizite Typumwandlung ....	80
<b>7</b>	<b>Ausdrücke, Operatoren und Operanden – Die Verarbeitung von Daten ...</b>	<b>83</b>
7.1	Was behandeln wir in diesem Kapitel? ....	83
7.2	Ausdrücke ....	83
7.3	Operationen mit Operatoren und Operanden. ....	84
7.3.1	Arithmetische Operatoren ....	84
7.3.2	Der String-Verkettungsoperator ....	87

7.3.3	Zuweisungsoperatoren . . . . .	88
7.3.4	Boolesche Operatoren (Vergleichsoperatoren) . . . . .	89
7.3.5	Logische Operatoren . . . . .	90
7.3.6	Die Membership-Operatoren . . . . .	92
7.3.7	Identitätsoperatoren . . . . .	92
7.3.8	Bitweise Operatoren. . . . .	93
7.4	Python-Sonderfälle bei der Auswertung . . . . .	98
7.4.1	Verkettete Auswertung . . . . .	98
7.4.2	Der Walross-Operator . . . . .	99
7.5	Operatorvorrang und Ausdrucksbewertung . . . . .	101
7.5.1	Die Priorität der Python-Operatoren . . . . .	101
7.5.2	Bewertung von Ausdrücken . . . . .	101
<b>8</b>	<b>Kontrollstrukturen – Die Steuerung des Programmflusses</b> . . . . .	<b>103</b>
8.1	Was behandeln wir in diesem Kapitel? . . . . .	103
8.2	Was sind Kontrollstrukturen? . . . . .	103
8.3	Die Kontrollstrukturen in Python. . . . .	104
8.3.1	Entscheidungsanweisungen . . . . .	104
8.3.2	Iterationsanweisungen . . . . .	113
8.3.3	Sprunganweisungen . . . . .	116
<b>9</b>	<b>Funktionen in Python – Modularisierung mit „Unterprogrammen“</b> . . . . .	<b>119</b>
9.1	Was behandeln wir in diesem Kapitel? . . . . .	119
9.2	In Python eigene Funktionen deklarieren – Das Schlüsselwort def. . . . .	119
9.2.1	Übergabewerte . . . . .	120
9.2.2	Rückgabewerte. . . . .	121
9.3	Funktionen aufrufen. . . . .	121
9.3.1	Stehen in Python global deklarierte Variablen in der Funktion zur Verfügung? . . . . .	123
9.4	Rekursion . . . . .	124
9.5	Innere Funktionen – Closures . . . . .	128
9.6	Lambda-Ausdrücke und anonyme Funktionen . . . . .	129
9.6.1	Lambda-Funktionen verwenden . . . . .	129
9.7	Besondere Situationen bei Funktionen in Python . . . . .	130
9.7.1	Lokale Variablen in Funktionen. . . . .	130
9.7.2	Die Anzahl der Parameter passt nicht . . . . .	131
9.7.3	Unerreichbarer Code . . . . .	134
<b>10</b>	<b>Sequenzielle Datenstrukturen – Mehrere Informationen gemeinsam verwalten</b> . . . . .	<b>137</b>
10.1	Was behandeln wir in diesem Kapitel? . . . . .	137
10.2	Was sind sequenzielle Datenstrukturen? . . . . .	137

10.2.1	Zeichenketten als sequenzielle Ansammlung von Zeichenliteralen . . . . .	138
10.2.2	Arrays. . . . .	138
10.3	Tupel . . . . .	139
10.3.1	Verschachtelte Tupel . . . . .	139
10.3.2	Tupel und der Membership-Operator. . . . .	140
10.3.3	Einzelne Einträge in Tupel ansprechen . . . . .	143
10.3.4	Die Anzahl der Elemente in einem Tupel bestimmen . . . . .	146
10.4	Dynamische Listen. . . . .	146
10.4.1	Warum Listen und Tupel? . . . . .	147
10.5	Methoden für Listen. . . . .	148
10.5.1	Verschiedene Listenmethoden in einem Beispiel . . . . .	149
10.5.2	Einen Stack erzeugen. . . . .	150
10.5.3	Eine Queue mit einer Liste erzeugen. . . . .	151
10.6	Dictionaries . . . . .	152
10.6.1	Spezielle Methoden für Dictionaries . . . . .	153
10.6.2	Ein Beispiel zum allgemeinen Umgang mit Dictionaries . . . . .	154
10.6.3	Ein Dictionary aktualisieren oder erweitern . . . . .	155
10.6.4	Iteration über ein Dictionary . . . . .	156
10.7	Mengen . . . . .	157
10.7.1	Vereinfachte Notation . . . . .	157
10.7.2	Operationen auf set-Objekten . . . . .	158
10.8	Operatoren bei sequenziellen Datentypen . . . . .	160
10.8.1	Der Plusoperator . . . . .	160
10.8.2	Multiplikationen mit sequenziellen Datentypen . . . . .	160
10.8.3	Inhalt überprüfen . . . . .	161
10.9	Über sequenzielle Strukturen iterieren. . . . .	163
10.10	Pattern-Matching . . . . .	164
10.11	Comprehensions. . . . .	169
10.12	Anwendung des Walrus Operator . . . . .	171
<b>11</b>	<b>Objektorientierte Programmierung in Python – Klassen, Objekte, Eigenschaften und Methoden. . . . .</b>	<b>173</b>
11.1	Was behandeln wir in diesem Kapitel? . . . . .	173
11.2	Hintergründe der OOP. . . . .	173
11.2.1	Ziele der OOP – Wiederverwendbarkeit und bessere Softwarequalität . . . . .	174
11.2.2	Kernkonzepte der Objektorientierung . . . . .	175
11.3	Klassen. . . . .	177
11.3.1	Klassen als Baupläne, Konstruktoren und Destruktoren . . . . .	177
11.3.2	Der konkrete Klassenaufbau in Python . . . . .	177
11.3.3	Die konkrete Instanziierung. . . . .	178

11.4	Details zu Objekten . . . . .	182
11.4.1	OO-Philosophie als Abstraktion . . . . .	183
11.4.2	Instanzelemente versus Klasselemente . . . . .	183
11.4.3	Der Aufbau von Objekten in Python . . . . .	184
11.4.4	Zugriff auf Objektbestandteile . . . . .	184
11.4.5	Von Grund auf objektorientiert . . . . .	189
11.5	Klassenmethoden und statische Methoden . . . . .	190
11.5.1	Klassenmethoden . . . . .	190
11.5.2	Statische Methoden . . . . .	191
11.6	Eine Frage der Sichtbarkeit . . . . .	193
11.6.1	Ein Beispiel für den Zugriff auf ein öffentliches Element . . .	193
11.6.2	Ein Beispiel für den versuchten Zugriff auf ein privates Element von außen . . . . .	194
11.6.3	Getter und Setter . . . . .	195
11.7	Ein Objekt löschen. . . . .	197
11.7.1	Ein Beispiel für das Redefinieren des Destruktors . . . . .	198
11.8	Ein paar besondere OO-Techniken . . . . .	198
11.8.1	Eine to-string-Funktionalität bereitstellen – <code>__str__</code> . . . . .	198
11.8.2	Objekte dynamisch erweitern, das Dictionary <code>__dict__</code> und Slots. . . . .	199
11.8.3	Dynamische Erzeugung von Klassen, Metaklassen und die Klasse <code>type</code> . . . . .	201
11.8.4	Shallow Copy: flaches und tiefes Kopieren . . . . .	202
11.8.5	Reference Counting . . . . .	204
11.9	Vererbung . . . . .	205
11.9.1	Grundlagentheorie zur Vererbung . . . . .	205
11.9.2	Umsetzung von Vererbung in Python. . . . .	206
11.9.3	Mehrfachvererbung in Python . . . . .	207
11.9.4	Polymorphie über Überschreiben und Überladen . . . . .	210
11.10	Was ist mit Schnittstellen und abstrakten Klassen in Python? . . . . .	212
11.10.1	Abstrakte Superklassen . . . . .	212
11.10.2	Was ist im Allgemeinen eine Schnittstelle? . . . . .	213
11.11	Module und Pakete. . . . .	213
11.11.1	Die <code>import</code> -Anweisung . . . . .	214
11.11.2	Importieren mit <code>from</code> . . . . .	215
11.11.3	Pakete . . . . .	215
11.11.4	Das Python-API . . . . .	217
11.11.5	Fremde Pakete nutzen . . . . .	218
12	<b>Exception-Handling – Ausnahmsweise</b> . . . . .	219
12.1	Was behandeln wir in diesem Kapitel? . . . . .	219
12.2	Was sind Ausnahmen? . . . . .	220

12.3	Warum ein Ausnahmekonzept? . . . . .	220
12.4	Konkrete Ausnahmebehandlung in Python . . . . .	221
12.4.1	Ein erstes Beispiel mit einfacher Ausnahmebehandlung . . . . .	221
12.4.2	Mehrere Ausnahmeblöcke . . . . .	223
12.4.3	Die finally-Anweisung . . . . .	223
12.4.4	Praktische Beispiele . . . . .	224
12.5	Standard Exceptions. . . . .	226
12.5.1	Die Reihenfolge bei mehreren Ausnahmetypen . . . . .	227
12.6	Der else-Block . . . . .	228
12.7	Ausnahmeobjekte auswerten . . . . .	229
12.8	Werfen von Ausnahmen mit raise . . . . .	230
12.9	Eigene Ausnahmeklassen definieren . . . . .	232
12.10	Die assert-Anweisung . . . . .	232
<b>13</b>	<b>String-Verarbeitung in Python – Programmierte Textverarbeitung . . . . .</b>	<b>233</b>
13.1	Was behandeln wir in diesem Kapitel? . . . . .	233
13.2	Typische String-Verarbeitungstechniken . . . . .	234
13.3	Das konkrete Vorgehen in Python . . . . .	234
13.3.1	String-Konstanten und die Format Specification Mini-Language . . . . .	234
13.3.2	String-Funktionen . . . . .	235
13.3.3	String-Methoden . . . . .	235
13.4	Umgang mit regulären Ausdrücken . . . . .	236
13.4.1	Was sind allgemein reguläre Ausdrücke? . . . . .	237
13.4.2	Wo setzt man reguläre Ausdrücke ein? . . . . .	237
13.4.3	Details zu Pattern . . . . .	238
13.4.4	Optionen für die Häufigkeit . . . . .	241
13.4.5	Die Umsetzung von regulären Ausdrücken in Python – das Modul re . . . . .	241
13.4.6	Die Match-Objekte. . . . .	244
13.4.7	Ein paar Beispiele mit regulären Ausdrücken . . . . .	246
<b>14</b>	<b>Datei-, Datenträger- und Datenbankzugriffe – Dauerhafte Daten. . . . .</b>	<b>249</b>
14.1	Was behandeln wir in diesem Kapitel? . . . . .	249
14.2	Datenströme für die Ein- und Ausgabe . . . . .	249
14.2.1	Das Öffnen und Schließen einer Datei. . . . .	250
14.2.2	Schreiben in eine Datei . . . . .	251
14.2.3	Auslesen aus einer Datei . . . . .	252
14.2.4	Die with-Anweisung nutzen . . . . .	253
14.2.5	Lese- und Schreibvorgänge absichern . . . . .	253
14.3	Allgemeine Datei- und Verzeichnisoperationen. . . . .	256

14.4	Objekte serialisieren und deserialisieren . . . . .	258
14.4.1	Mit dump() den Objektzustand persistent machen . . . . .	259
14.4.2	Mit load() den Objektzustand reproduzieren . . . . .	259
14.5	Datenbankzugriffe . . . . .	260
14.5.1	Was ist SQLite? . . . . .	260
14.5.2	Zugriff auf SQLite in Python . . . . .	261
14.5.3	Ein konkretes Datenbankbeispiel . . . . .	262
<b>15</b>	<b>Umgang mit Datum und Zeit – Terminsachen . . . . .</b>	<b>265</b>
15.1	Was behandeln wir in diesem Kapitel? . . . . .	265
15.2	Allgemeines zum Umgang mit Datum und Zeit . . . . .	265
15.3	Die Python-Module . . . . .	266
15.4	Typische Beispiele für Operationen mit Zeit und Datum . . . . .	266
15.4.1	Das aktuelle Systemdatum des Computers auslesen . . . . .	266
15.4.2	Ein beliebiges Datumsobjekt erstellen . . . . .	267
<b>16</b>	<b>Grafische Oberflächen (GUI) mit Python – tkinter &amp; Co als</b>	
	<b>GUI-Framework . . . . .</b>	<b>271</b>
16.1	Was behandeln wir in diesem Kapitel? . . . . .	271
16.2	Hintergrundinformationen zu modernen grafischen Oberflächen . . . . .	271
16.3	Konkrete GUI-Konzepte in Python und das Modul tkinter . . . . .	272
16.3.1	Ein Fenster vom Typ TK als Basis jeder GUI-Applikation . . . . .	272
16.3.2	Der übliche OO-Ansatz . . . . .	272
16.3.3	Die Layout-Manager . . . . .	273
16.3.4	Wichtige GUI-Elemente . . . . .	280
16.4	Die Ereignisbehandlung . . . . .	281
16.4.1	Die konkrete Ereignisbehandlung in Python . . . . .	281
16.4.2	Lambda-Ausdrücke verwenden . . . . .	283
16.5	Eine grafische Datenbankapplikation . . . . .	284
16.6	PyQt als Alternative . . . . .	288
	<b>Stichwortverzeichnis . . . . .</b>	<b>291</b>