

Inhaltsverzeichnis

Vorwort	xv
 I Einstieg	 1
1 Einführung	3
1.1 Python im Überblick	3
1.2 Los geht's – Installation	6
1.2.1 Python-Download	6
1.2.2 Installation von Python	7
1.2.3 Nacharbeiten nach der Python-Installation	8
1.2.4 Python-Installation prüfen	9
1.2.5 Python-Programm als Skript ausführen	9
1.3 Entwicklungsumgebungen	10
1.3.1 Installation von PyCharm	11
1.3.2 PyCharm starten	13
1.3.3 Erstes Projekt in PyCharm	14
1.3.4 Erstes Modul in PyCharm	15
 2 Schnelleinstieg	 19
2.1 Hallo Welt (Hello World)	19
2.2 Variablen und Datentypen	20
2.2.1 Definition von Variablen	20
2.2.2 Variablen und Typen	22
2.2.3 Bezeichner (Variablennamen)	23
2.3 Operatoren im Überblick	24
2.3.1 Arithmetische Operatoren	24
2.3.2 Zuweisungsoperatoren	26
2.3.3 Vergleichsoperatoren	28
2.3.4 Logische Operatoren	29
2.4 Fallunterscheidungen	30

2.5	Funktionen	32
2.5.1	Eigene Funktionen definieren	33
2.5.2	Nützliche Beispiele aus Python	35
2.5.3	Fehlerbehandlung und Exceptions	36
2.6	Kommentare	36
2.7	Module	37
2.8	Built-in-Datentypen	39
2.8.1	List	39
2.8.2	Tupel	39
2.8.3	Set	40
2.8.4	Dict	41
2.9	Schleifen	42
2.9.1	Besonderheit: Ranges	42
2.9.2	Indexbasierte <code>for-in</code> -Schleife	42
2.9.3	Wertebasierte <code>for-in</code> -Schleife	43
2.9.4	Index- und wertebasierte <code>for-in-enumerate</code> -Schleife	44
2.9.5	Die <code>while</code> -Schleife	45
2.10	Weiterführende Informationen	46
2.11	Aufgaben und Lösungen	47
2.11.1	Aufgabe 1: Mathematische Berechnungen	47
2.11.2	Aufgabe 2: Bedingung vereinfachen	47
2.11.3	Aufgabe 3: Funktion und <code>if</code>	48
2.11.4	Aufgabe 4: Selbstabholerrabatt	48
2.11.5	Aufgabe 5: Schleifen mit Berechnungen	49
2.11.6	Aufgabe 6: Schleifen und fixe Schrittweite	50
2.11.7	Aufgabe 7: Schleifen mit variabler Schrittweite	50
2.11.8	Aufgabe 8: Verschachtelte Schleifen – Variante 1	51
2.11.9	Aufgabe 9: Verschachtelte Schleifen – Variante 2	52
2.11.10	Aufgabe 10: Verschachtelte Schleifen – Variante 3	53
3	Strings	55
3.1	Schnelleinstieg	55
3.1.1	Gebräuchliche Stringaktionen	55
3.1.2	Suchen und Ersetzen	60
3.1.3	Informationen extrahieren und formatieren	61
3.1.4	Praxisrelevante Funktionen im Kurzüberblick	63
3.2	Nächste Schritte	64
3.2.1	Zeichenverarbeitung	64
3.2.2	Strings und Listen	64
3.2.3	Mehrzeilige Strings	66

3.3	Aufgaben und Lösungen	68
3.3.1	Aufgabe 1: Länge, Zeichen und Enthaltensein	68
3.3.2	Aufgabe 2: Zeichen wiederholen	68
3.3.3	Aufgabe 3: Vokale raten	69
3.3.4	Aufgabe 4: String Merge	71
4	Klassen und Objektorientierung	73
4.1	Schnelleinstieg	73
4.1.1	Grundlagen zu Klassen und Objekten	74
4.1.2	Eigenschaften (Attribute)	77
4.1.3	Verhalten (Methoden)	79
4.1.4	Objekte vergleichen – die Rolle von <code>__eq__()</code>	81
4.2	Nächste Schritte	84
4.2.1	Klassen ausführbar machen	84
4.2.2	Imports und Packages	86
4.2.3	Übergang zum Einsatz einer IDE	87
4.2.4	Verstecken von Informationen	90
4.2.5	Packages: Auswirkungen auf unsere Applikation	94
4.3	Vererbung	97
4.3.1	Basisklassen	98
4.3.2	Typprüfung mit <code>isinstance()</code>	99
4.3.3	Generalisierung und Spezialisierung	100
4.3.4	Polymorphie	101
4.4	Aufgaben und Lösungen	104
4.4.1	Aufgabe 1: Superheld	104
4.4.2	Aufgabe 2: Zähler	105
4.4.3	Aufgabe 3: Zähler mit Überlauf	107
5	Collections	111
5.1	Schnelleinstieg	111
5.1.1	Die Klasse <code>list</code>	111
5.1.2	Die Klasse <code>set</code>	117
5.1.3	Die Klasse <code>dict</code>	120
5.2	Nächste Schritte	124
5.2.1	Comprehensions	124
5.2.2	Slicing – Zugriff auf Teilbereiche	125
5.2.3	Sortierung – <code>sort()</code> / <code>sorted()</code>	126
5.2.4	Tauschen von Elementen – <code>swap()</code>	128
5.2.5	Reihenfolge umkehren – <code>reverse()</code> und <code>reversed()</code>	131
5.2.6	Mehrdimensionale Listen	132

5.3	Praxisbeispiel: Einen Stack selbst realisieren	134
5.3.1	Implementierung	134
5.3.2	Stack im Einsatz	136
5.4	Praxisbeispiel: Flächen füllen	137
5.5	Aufgaben und Lösungen	139
5.5.1	Aufgabe 1: Tennisverein-Mitgliederliste	139
5.5.2	Aufgabe 2: Liste mit Farbnamen füllen und filtern	140
5.5.3	Aufgabe 3: Duplikate entfernen – Variante 1	140
5.5.4	Aufgabe 4: Duplikate entfernen – Variante 2	141
5.5.5	Aufgabe 5: Hauptstädte	142
5.5.6	Aufgabe 6: Häufigkeiten von Namen	143
5.5.7	Aufgabe 7: Objekte mit Dictionary selbst gebaut	144
5.5.8	Aufgabe 8: Rotation um eine oder mehrere Positionen	145
5.5.9	Aufgabe 9: Dreieckige Liste: Upside Down	147
6	Ergänzendes Wissen	149
6.1	Benutzereingaben <code>input()</code>	149
6.2	Zufallswerte und das Modul <code>random</code>	150
6.3	Besonderheiten von Parametern	152
6.3.1	Normale Parameter mit Position bzw. Name	152
6.3.2	Parameter mit Defaultwert	153
6.3.3	Var Args – variable Anzahl an Argumenten	154
6.4	Ternary-Operator	156
6.5	Aufzählungen mit <code>Enum</code>	157
6.6	<code>break</code> , <code>continue</code> und <code>else</code> in Schleifen	160
6.6.1	Funktionsweise von <code>break</code> und <code>continue</code>	160
6.6.2	Wie macht man es besser?	162
6.6.3	Besonderheit: <code>else</code> in Schleifen	164
6.7	Ausdrücke mit <code>eval()</code> auswerten	165
6.8	Rekursion	166
6.9	Aufgaben und Lösungen	169
6.9.1	Aufgabe 1: Würfelspiel	169
6.9.2	Aufgabe 2: Temperaturumrechnung	170
6.9.3	Aufgabe 3: Palindrom-Prüfung mit Rekursion	171
6.9.4	Aufgabe 4: Einarmiger Bandit	172

II	Aufstieg	173
7	Collections Advanced	175
7.1	Sequenzielle Datentypen	175
7.2	Iteratoren	176
7.3	Generatoren	179
7.4	Datencontainer mit <code>namedtuple</code>	182
7.5	Einstieg in Lambdas	184
7.5.1	Syntax von Lambdas	184
7.5.2	Lambdas im Einsatz mit <code>filter()</code> , <code>map()</code> und <code>reduce()</code>	185
7.5.3	Lambdas im Einsatz mit <code>sort()</code>	188
7.5.4	Lambdas im Einsatz mit <code>groupby()</code>	189
7.6	Aufgaben und Lösungen	192
7.6.1	Aufgabe 1: Obstkorb	192
7.6.2	Aufgabe 2: Erwachsene aus Personenliste extrahieren	192
7.6.3	Aufgabe 3: Eigene Implementierung von <code>rindex()</code>	193
7.6.4	Aufgabe 4: Elemente eines Dictionaries allgemeingültig filtern	194
7.6.5	Aufgabe 5: Every-N-th-Iterator	196
7.6.6	Aufgabe 6: Greeting-Generator	198
7.6.7	Aufgabe 7: Fibonacci-Generator	199
7.6.8	Aufgabe 8: Sortieren und Gruppieren	200
8	Verarbeitung von Dateien	203
8.1	Schnelleinstieg	203
8.1.1	Anlegen von Dateien und Verzeichnissen	204
8.1.2	Aktuelles Verzeichnis wechseln	204
8.1.3	Aktuelles Verzeichnis und absoluten Pfad ermitteln	205
8.1.4	Inhalt eines Verzeichnisses auflisten	205
8.1.5	Pfad ist Datei oder Verzeichnis?	205
8.1.6	Auf Existenz prüfen	206
8.1.7	Informationen in Dateien schreiben und daraus lesen	206
8.1.8	Einfluss der Verarbeitungsmodi	210
8.1.9	Diverse Informationen ermitteln	211
8.1.10	Kopieren	212
8.1.11	Umbenennen	213
8.1.12	Löschen	214
8.2	Praxisbeispiel: Directory-Baum darstellen	215
8.2.1	Basisvariante	215
8.2.2	Variante mit schönerer Darstellung	216
8.2.3	Finale Variante mit ausgeklügelter Darstellung	217

8.3	JSON-Verarbeitung	218
8.3.1	JSON in eine Datei schreiben	218
8.3.2	Lesen von JSON aus einer Datei	219
8.3.3	Pretty Printing	220
8.4	Aufgaben und Lösungen	221
8.4.1	Aufgabe 1: Texte in Datei schreiben und wieder lesen	221
8.4.2	Aufgabe 2: Dateigrößen	221
8.4.3	Aufgabe 3: Existenzprüfung	222
8.4.4	Aufgabe 4: Rechteprüfung	223
8.4.5	Aufgabe 5: Verzeichnisinhalt auflisten	223
9	Fehlerbehandlung mit Exceptions	225
9.1	Schnelleinstieg	225
9.1.1	Fehlerbehandlung	226
9.1.2	Exceptions selbst auslösen – <code>raise</code>	232
9.1.3	Eigene Exception-Typen definieren	233
9.1.4	Propagation von Exceptions	234
9.2	Fehlerbehandlung in der Praxis	236
9.2.1	Elegante Prüfungen mit <code>assert</code>	237
9.3	Automatic Resource Management (<code>with</code>)	239
9.4	Aufgaben und Lösungen	240
9.4.1	Aufgabe 1: Abgesicherter Indexzugriff – Kür mit Fallback-Wert	240
9.4.2	Aufgabe 2: Einfacher Taschenrechner	241
9.4.3	Aufgabe 3: Resource Handling	242
10	Datumsverarbeitung	245
10.1	Schnelleinstieg	245
10.1.1	Zeitpunkte und die Klasse <code>datetime</code>	245
10.1.2	Datumswerte und die Klasse <code>date</code>	247
10.1.3	Zeit und die Klasse <code>time</code>	250
10.1.4	Zeitdifferenzen und die Klasse <code>timedelta</code>	251
10.1.5	Berechnungen	252
10.1.6	Formatierung und Parsing	253
10.2	Praxisbeispiel: Kalenderausgabe	255
10.3	Aufgaben und Lösungen	258
10.3.1	Aufgabe 1: Wochentage	258
10.3.2	Aufgabe 2: Freitag, der 13.	259
10.3.3	Aufgabe 3: Mehrmals Freitag, der 13.	260
10.3.4	Aufgabe 4: Schaltjahre	261

III	Praxisbeispiele	263
11	Praxisbeispiel: Tic Tac Toe	265
11.1	Spiefeld initialisieren und darstellen	265
11.2	Setzen der Steine	266
11.3	Prüfen auf Sieg	267
11.4	Bausteine im Einsatz	268
12	Praxisbeispiel: CSV-Highscore-Liste einlesen	271
12.1	Verarbeitung von Spielständen (Highscores)	271
12.2	Extraktion der Daten	272
12.3	Besonderheiten der Implementierung	273
13	Praxisbeispiel: Worträtsel	275
13.1	Applikationsdesign – Vorüberlegungen zur Strukturierung	276
13.2	Einlesen der verfügbaren Wörter	276
13.3	Hilfsdatenstrukturen	278
13.4	Datenmodell	279
13.4.1	Datenspeicherung und Initialisierung	279
13.4.2	Zufällige Wahl von Richtung, Position, Wort und Buchstabe	280
13.4.3	Algorithmus zum Verstecken von Wörtern	280
13.4.4	Wort prüfen und platzieren	281
13.5	HTML-Erzeugung	282
13.6	Ausgabe als HTML und Darstellung im Browser	284
13.7	Hauptapplikation	284
13.8	Fazit	286
IV	Schlussgedanken	287
14	Gute Angewohnheiten	289
14.1	Grundregeln eines guten Programmierstils	289
14.1.1	Keep It Human-Readable	289
14.1.2	Keep it Understandable	289
14.2	Coding Conventions	292
14.2.1	PEP 8 – Coding Standard	292
14.2.2	Zen of Python	294
14.2.3	Namensgebung	294
14.2.4	Dokumentation	297
14.2.5	Programmdesign	297
14.2.6	Parameterlisten	298
14.2.7	Logik und Kontrollfluss	298

14.3 Auch ans Testen denken	299
14.3.1 Das Pytest-Framework	299
14.3.2 Schreiben und Ausführen von Tests	300
15 Schlusswort	303
 V Anhang	 305
 A Schlüsselwörter im Überblick	 307
A.1 Schlüsselwörter im Überblick	307
 B Schnelleinstieg Python-REPL	 311
B.1 Python-REPL	311
 C Wesentliche Neuerungen aus Python 3.10 im Kurzüberblick	 315
C.1 Fehlermeldungen	315
C.1.1 Fehlermeldungen bei Zuweisungen	315
C.1.2 Fehlermeldungen bei unvollständigen Strings	316
C.2 Fallunterscheidungen mit <code>match</code>	317
C.3 Verschiedenes	319
C.3.1 Verbesserungen bei Kontextmanagern	319
C.3.2 Verbesserungen der Performance	319
C.3.3 Erweiterung bei <code>zip()</code>	320
C.3.4 Verbesserungen bei Typprüfungen	321
 Literaturverzeichnis	 323
 Index	 325