

Inhaltsverzeichnis

| | | |
|----------|--|-----------|
| | Einleitung | 17 |
| | Python in Studium und Ausbildung | 17 |
| | Der Aufbau des Buchs | 17 |
| | Achten Sie auf den Schrifttyp! | 18 |
| | Kostenloses E-Book | 18 |
| | Programmtexte und Lösungen zum Download | 19 |
| 1 | Willkommen zu Python! | 21 |
| 1.1 | Die Programmiersprache Python | 21 |
| 1.2 | Was ist ein Algorithmus? | 22 |
| 1.3 | Syntax und Semantik | 22 |
| 1.4 | Interpreter und Compiler | 23 |
| 1.5 | Python installieren | 23 |
| 1.6 | Python im interaktiven Modus | 25 |
| 1.7 | Die Entwicklungsumgebung IDLE | 26 |
| 1.8 | Hotkeys für die IDLE-Shell | 27 |
| 1.9 | Anweisungen | 27 |
| | 1.9.1 Ausdruck | 27 |
| | 1.9.2 Funktionsaufruf | 28 |
| | 1.9.3 Zuweisung | 28 |
| | 1.9.4 Erweiterte Zuweisungen | 31 |
| 1.10 | Zahlen verarbeiten – Python als Taschenrechner | 32 |
| | 1.10.1 Operatoren | 32 |
| | 1.10.2 Variablen verwenden | 34 |
| 1.11 | Eine weitere Entwicklungsumgebung: Thonny | 34 |
| 1.12 | Notebooks mit Jupyter und CoLab | 36 |
| 1.13 | Rückblick | 36 |
| 1.14 | Übungen | 37 |
| 1.15 | Lösung der Frage: Semantik im Alltag | 38 |
| 2 | Datentypen – die Python-Typ-Hierarchie | 39 |
| 2.1 | Literale und die Funktion type() | 39 |
| 2.2 | Die Python-Typ-Hierarchie | 40 |

| | | |
|----------|--|-----------|
| 2.3 | Standard-Typen | 40 |
| 2.3.1 | Ganze Zahl (int) | 40 |
| 2.3.2 | Gleitkommazahl (float) | 42 |
| 2.3.3 | Komplexe Zahlen (complex) | 42 |
| 2.3.4 | Zeichenkette (str) | 43 |
| 2.3.5 | Tupel (tuple) | 44 |
| 2.3.6 | Liste (list) | 44 |
| 2.3.7 | Menge (set) | 44 |
| 2.3.8 | Dictionary (dict) | 45 |
| 2.3.9 | Wahrheitswerte – der Datentyp bool | 45 |
| 2.3.10 | NoneType | 46 |
| 2.4 | Gemeinsame Operationen für Kollektionen | 46 |
| 2.4.1 | Kollektion | 47 |
| 2.4.2 | Sequenz | 47 |
| 2.5 | Objekte eines Typs erzeugen – Casting | 48 |
| 2.6 | Dynamische Typisierung | 50 |
| 2.7 | Rückblick | 50 |
| 2.8 | Übung: Anweisungen | 51 |
| 3 | Interaktive Programme | 53 |
| 3.1 | Das erste Python-Skript | 53 |
| 3.2 | Das EVA-Prinzip | 55 |
| 3.3 | Kommentare | 57 |
| 3.4 | Projekt: Volumenberechnung | 58 |
| 3.4.1 | Kürzerer Programmtext durch verschachtelte Funktionsaufrufe | 59 |
| 3.4.2 | Aufruf der Funktion print() mit mehreren Argumenten ... | 60 |
| 3.5 | Python-Programme starten | 60 |
| 3.5.1 | Ausführung auf der Kommandozeile | 61 |
| 3.5.2 | Start durch Anklicken des Programm-Icons unter Windows | 62 |
| 3.5.3 | Python-Programme unter Linux – die Shebang-Zeile | 64 |
| 3.5.4 | Starten im Finder von macOS | 64 |
| 3.6 | Fehler finden | 64 |
| 3.6.1 | Syntaxfehler | 64 |
| 3.6.2 | Laufzeitfehler | 65 |
| 3.6.3 | Semantische Fehler | 66 |
| 3.6.4 | Tipps zum Fehlerfinden | 66 |
| 3.7 | Rückblick | 67 |
| 3.8 | Übungen | 67 |
| 3.9 | Lösungen zu den Fragen | 69 |

| | | |
|----------|--|-----------|
| 4 | Kontrollstrukturen | 71 |
| 4.1 | Programmverzweigungen | 71 |
| 4.1.1 | Einseitige Verzweigung (if) | 71 |
| 4.1.2 | Projekt: Passwort | 72 |
| 4.1.3 | Zweiseitige Verzweigung (if...else) | 73 |
| 4.1.4 | Projekt: Kinokarte | 74 |
| 4.1.5 | Fallunterscheidung (if...elif...else) | 75 |
| 4.1.6 | Projekt: Auskunftautomat | 76 |
| 4.2 | Das Layout von Python-Programmen: Zeilen und Blöcke | 77 |
| 4.2.1 | Block | 77 |
| 4.2.2 | Zeilenstruktur | 77 |
| 4.3 | Bedingungen konstruieren | 78 |
| 4.3.1 | Boolesche Werte | 79 |
| 4.3.2 | Boolesche Operatoren | 79 |
| 4.3.3 | Vergleichsketten | 80 |
| 4.3.4 | Projekt: Früchte erkennen | 80 |
| 4.4 | Bedingte Wiederholung – while | 82 |
| 4.4.1 | Projekt: Aufsummieren | 83 |
| 4.4.2 | Projekt: Planeten | 84 |
| 4.4.3 | Projekt: Wurzelberechnung (Mathematik) | 85 |
| 4.4.4 | Endloswiederholung | 87 |
| 4.5 | Iterationen – for | 87 |
| 4.5.1 | Wiederholungen mit range() | 89 |
| 4.6 | Rückblick | 90 |
| 4.7 | Übungen | 90 |
| 4.8 | Lösungen zu den Fragen | 92 |
| 5 | Funktionen | 93 |
| 5.1 | Warum definiert man Funktionen? | 93 |
| 5.2 | Definition und Aufruf einer Funktion | 93 |
| 5.2.1 | Projekt: Fallhöhe berechnen | 94 |
| 5.3 | Optionale Parameter und voreingestellte Werte | 96 |
| 5.3.1 | Erweiterung des Projekts: Fallhöhe auf unterschiedlichen Himmelskörpern | 96 |
| 5.4 | Eine Funktion in der Shell testen | 97 |
| 5.5 | Die return-Anweisung | 98 |
| 5.5.1 | Prozeduren | 98 |
| 5.5.2 | Wirkungen der return-Anweisung | 98 |
| 5.6 | Positionsargumente und Schlüsselwortargumente | 99 |

| | | |
|----------|--|------------|
| 5.7 | Guter Programmierstil | 101 |
| 5.7.1 | Funktionsname | 101 |
| 5.7.2 | Funktionsannotationen | 101 |
| 5.7.3 | Docstring | 101 |
| 5.7.4 | Signatur | 102 |
| 5.8 | Die print()-Funktion unter der Lupe | 103 |
| 5.9 | Globale und lokale Namen | 104 |
| 5.10 | Rekursive Funktionen | 105 |
| 5.10.1 | Projekt: Die Berechnung der Fakultät | 105 |
| 5.11 | Lambda-Funktionen * | 107 |
| 5.12 | Funktionen als Argumente: map() und filter() * | 108 |
| 5.12.1 | Mapping | 108 |
| 5.12.2 | Filtern | 110 |
| 5.13 | Rückblick | 111 |
| 5.14 | Übungen | 111 |
| 5.15 | Lösungen zu den Fragen | 112 |
| 6 | Mit Modulen arbeiten | 115 |
| 6.1 | Importanweisungen | 115 |
| 6.1.1 | Ein Modul importieren | 115 |
| 6.1.2 | Funktionen aus einem Modul importieren | 116 |
| 6.2 | Mathematische Funktionen: Das Modul math | 117 |
| 6.3 | Zufallsfunktionen: Das Modul random | 118 |
| 6.3.1 | Projekt: Würfeln | 118 |
| 6.3.2 | Projekt: Wer ist der Nächste? | 119 |
| 6.4 | Datum und Zeit | 119 |
| 6.4.1 | Projekt: Uhrzeit | 120 |
| 6.4.2 | Projekt: Rechentrainer | 120 |
| 6.5 | Ein eigenes Modul erstellen | 122 |
| 6.5.1 | Projekt: Ein Modul zur Volumenberechnung | 122 |
| 6.5.2 | Welchen Vorteil haben Module? | 126 |
| 6.6 | Module aus dem Python Package Index (PyPI) | 126 |
| 6.7 | Rückblick | 126 |
| 6.8 | Übungen | 127 |
| 7 | Mit Kollektionen modellieren | 129 |
| 7.1 | Sequenzen | 129 |
| 7.1.1 | Listen | 129 |
| 7.1.2 | Tupel | 130 |
| 7.1.3 | Komplexe Sequenzen | 130 |
| 7.1.4 | Iteration über eine Liste aus Tupeln | 131 |
| 7.1.5 | Gemeinsame Operationen für Sequenzen | 132 |

| | | |
|----------|---|------------|
| 7.1.6 | Spezielle Operationen für Listen | 133 |
| 7.1.7 | Sortieren | 134 |
| 7.1.8 | Eine Liste erzeugen | 135 |
| 7.2 | Projekt: Telefonliste | 137 |
| 7.3 | Dictionaries | 139 |
| 7.3.1 | Operationen für Dictionaries | 140 |
| 7.3.2 | Ein Dictionary ändern | 140 |
| 7.4 | Projekt: Vokabeltrainer | 141 |
| 7.5 | Projekt: Routenplaner | 143 |
| 7.5.1 | Verkehrswege und Graphen | 143 |
| 7.5.2 | Programmierung | 145 |
| 7.6 | Rückblick | 147 |
| 7.7 | Übungen | 147 |
| 7.8 | Lösungen zu den Fragen | 148 |
| 8 | Daten speichern | 151 |
| 8.1 | Wie werden Daten gespeichert? | 151 |
| 8.1.1 | Dateien öffnen | 151 |
| 8.1.2 | Stream-Methoden | 152 |
| 8.1.3 | Texte speichern und laden | 153 |
| 8.1.4 | Binärdateien und Bytestrings | 153 |
| 8.1.5 | Pfade im Verzeichnisbaum | 154 |
| 8.2 | Laufzeitfehler abfangen | 155 |
| 8.2.1 | try...except | 156 |
| 8.2.2 | try...except...finally | 156 |
| 8.3 | with-Anweisungen | 157 |
| 8.4 | Projekt: Logbuch | 158 |
| 8.5 | Datenstrukturen speichern und laden: Das Modul pickle | 160 |
| 8.5.1 | Speichern | 160 |
| 8.5.2 | Laden | 161 |
| 8.6 | Projekt: Digitaler Planer | 162 |
| 8.7 | Daten im JSON-Format speichern | 165 |
| 8.7.1 | Aufbau eines JSON-Texts | 165 |
| 8.7.2 | Die Grenzen von JSON | 167 |
| 8.8 | Projekt: Temperaturdaten | 167 |
| 8.8.1 | Writer | 167 |
| 8.8.2 | Reader | 168 |
| 8.9 | Daten aus dem Internet | 169 |
| 8.10 | Projekt: Digitale Bibliothek | 170 |
| 8.11 | Rückblick | 172 |
| 8.12 | Übung: News-Check | 172 |
| 8.13 | Lösungen zu den Fragen | 173 |

| | | |
|-----------|--|------------|
| 9 | Textverarbeitung | 175 |
| 9.1 | Unicode-Nummern für Zeichen | 175 |
| 9.2 | Escape-Sequenzen | 176 |
| 9.3 | Stringmethoden | 177 |
| 9.4 | Projekt: Goethes Wortschatz | 179 |
| 9.5 | Projekt: Wie warm wird es heute? | 180 |
| 9.6 | Ausblick: Reguläre Ausdrücke * | 182 |
| 9.6.1 | Was ist ein regulärer Ausdruck? | 182 |
| 9.6.2 | Aufbau eines regulären Ausdrucks | 183 |
| 9.6.3 | Textpassagen finden mit findall() | 184 |
| 9.6.4 | Platzhalter für Zeichen aus einer Zeichenmenge | 186 |
| 9.6.5 | Reguläre Ausdrücke verknüpfen | 187 |
| 9.6.6 | Quantoren | 187 |
| 9.6.7 | Sonderzeichen maskieren | 188 |
| 9.6.8 | Gieriges oder nicht gieriges Finden | 189 |
| 9.6.9 | Webscraping mit regulären Ausdrücken | 190 |
| 9.7 | Texte mit variablen Teilen | 191 |
| 9.7.1 | Formatierung | 191 |
| 9.7.2 | Platzhalter mit Namen | 192 |
| 9.7.3 | Formatangaben für Platzhalter | 192 |
| 9.8 | Projekt: Textanalyse | 193 |
| 9.9 | Projekt: Storytelling | 195 |
| 9.10 | Rückblick | 196 |
| 9.11 | Übungen | 196 |
| 9.12 | Lösungen zu den Fragen | 198 |
| 10 | Zugriff auf die Systemumgebung | 201 |
| 10.1 | Schnittstelle zum Betriebssystem: Das Modul os | 201 |
| 10.2 | Suchen und Eigenschaften ermitteln | 202 |
| 10.2.1 | Unterverzeichnisse ausgeben | 202 |
| 10.2.2 | Verzeichnisbaum durchlaufen | 203 |
| 10.3 | Dateien und Verzeichnisse anlegen und umbenennen | 206 |
| 10.3.1 | Projekt: Bilddateien umbenennen | 206 |
| 10.4 | Das Modul sys – die Schnittstelle zum Laufzeitsystem | 208 |
| 10.4.1 | Informationen über die aktuelle Systemumgebung abfragen | 208 |
| 10.4.2 | Kommandozeilenargumente abfragen | 209 |
| 10.4.3 | Blick hinter die Kulissen: Speicherverwaltung * | 210 |
| 10.4.4 | Zugriff auf Module | 212 |
| 10.4.5 | Die Standardausgabe in eine Datei umleiten | 213 |

| | | |
|-----------|---|------------|
| 10.5 | Rückblick | 214 |
| 10.6 | Übungen | 215 |
| 10.7 | Lösung zu der Frage: Welcher Kommentar passt? | 216 |
| 11 | Grafische Benutzungsoberflächen | 217 |
| 11.1 | Widgets | 217 |
| 11.2 | Das Anwendungsfenster Tk | 218 |
| 11.3 | Ein Widget einfügen | 219 |
| 11.4 | Das Aussehen der Widgets gestalten | 220 |
| | 11.4.1 Die Größe eines Widgets | 222 |
| 11.5 | Gemeinsame Methoden der Widgets | 222 |
| 11.6 | Schaltflächen und Eventhandler | 223 |
| | 11.6.1 Projekt: Motivator | 223 |
| 11.7 | Das Layout verfeinern | 224 |
| 11.8 | Raster-Layout | 227 |
| 11.9 | Projekt: 25 Farben – ein automatisches Farbfelder-Bild | 228 |
| 11.10 | Widgets zur Texteingabe | 230 |
| | 11.10.1 Einzeilige Eingabe: Das Entry-Widget | 230 |
| | 11.10.2 Mehrzeilige Eingabe: Das Text-Widget | 231 |
| | 11.10.3 Projekt: Reimen mit Goethe | 233 |
| 11.11 | Radiobuttons | 235 |
| | 11.11.1 Projekt: Währungsrechner | 236 |
| 11.12 | Dialogboxen | 237 |
| | 11.12.1 Projekt: Texteditor | 238 |
| 11.13 | Parallele Abläufe: Threads | 239 |
| | 11.13.1 Ein Experiment: Countdown | 240 |
| | 11.13.2 Eine Funktion in einem eigenen Thread ausführen | 241 |
| 11.14 | Rückblick | 242 |
| 11.15 | Übungen | 243 |
| 11.16 | Lösungen zu den Fragen | 245 |
| 12 | Grafik programmieren | 247 |
| 12.1 | Bilder auf Schaltflächen und Labels | 247 |
| | 12.1.1 Projekt: Würfelspiel | 247 |
| | 12.1.2 Bilder verändern | 249 |
| | 12.1.3 Projekt: Graustufen | 250 |
| 12.2 | Canvas | 252 |
| | 12.2.1 Flächen gestalten | 252 |
| | 12.2.2 Linien gestalten | 254 |
| | 12.2.3 ID-Nummern: Elemente löschen oder bewegen | 254 |
| 12.3 | Projekt: Creative Coding | 255 |

| | | |
|-----------|---|------------|
| 12.4 | Die Python Imaging Library (PIL) | 257 |
| 12.4.1 | Ein Image-Objekt aus einer Datei gewinnen | 258 |
| 12.4.2 | Ein Image-Objekt ohne Datei erzeugen | 259 |
| 12.4.3 | Attribute und Methoden von Image-Objekten | 259 |
| 12.4.4 | Bilder über Listen verarbeiten | 261 |
| 12.4.5 | Bilder einfügen | 263 |
| 12.4.6 | Projekt: Greenscreen | 263 |
| 12.4.7 | PIL.Image-Objekte in tkinter-Anwendungen | 265 |
| 12.4.8 | Projekt: Webcam-Viewer | 266 |
| 12.5 | Rückblick | 267 |
| 12.6 | Übungen | 268 |
| 13 | Fehler finden und vermeiden | 271 |
| 13.1 | Zusicherungen | 271 |
| 13.2 | Tracing | 273 |
| 13.2.1 | Beispiel: Quicksort | 273 |
| 13.3 | Debugging mit IDLE | 275 |
| 13.3.1 | Der Debugger der Python-Shell | 275 |
| 13.3.2 | Das Programm schrittweise durchlaufen | 276 |
| 13.3.3 | Haltepunkte setzen | 277 |
| 13.4 | Debugging mit Thonny | 278 |
| 13.5 | Rückblick | 280 |
| 13.6 | Lösungen zu den Fragen | 281 |
| 14 | Objektorientierte Programmierung | 283 |
| 14.1 | Klassen und Objekte | 283 |
| 14.1.1 | Was ist Objektorientierung? | 283 |
| 14.1.2 | Klassen entwerfen und grafisch darstellen – UML | 284 |
| 14.1.3 | Definition einer Klasse | 285 |
| 14.1.4 | Objekte einer Klasse erzeugen: Instanziierung | 286 |
| 14.1.5 | Auf Attribute zugreifen | 286 |
| 14.1.6 | Methoden aufrufen | 287 |
| 14.1.7 | Objekte mit variablen Anfangswerten | 287 |
| 14.1.8 | Metaphern in der Programmierung | 288 |
| 14.2 | Projekt: Geld | 288 |
| 14.2.1 | Mit Geld-Objekten rechnen | 289 |
| 14.2.2 | Klassenattribute | 290 |
| 14.2.3 | Operatoren überladen – Polymorphie | 290 |
| 14.3 | Magische Methoden | 293 |
| 14.4 | Projekt: Abrechnung | 294 |

| | | |
|-----------|---|------------|
| 14.5 | Vererbung | 296 |
| 14.6 | Projekt: Farbtester | 298 |
| 14.7 | Projekt: Zahlenregen | 301 |
| 14.8 | Rückblick | 305 |
| 14.9 | Übungen | 305 |
| 14.10 | Lösungen zu den Fragen | 308 |
| 15 | Datenbanktechnik | 309 |
| 15.1 | Was ist ein Datenbanksystem? | 309 |
| 15.2 | Eine Datenbank entwerfen – das Entity-Relationship-Diagramm (ER) | 309 |
| 15.3 | Relationale Datenbanken | 311 |
| 15.4 | Relationen mit Python darstellen * | 313 |
| | 15.4.1 Menge von Tupeln | 313 |
| | 15.4.2 Benannte Tupel (named tuples) | 313 |
| 15.5 | Das Modul sqlite3 – Schnittstelle zu einer SQL-Datenbank | 314 |
| | 15.5.1 Mit SQL Tabellen anlegen und Tupel eintragen | 315 |
| | 15.5.2 Mit sqlite3 eine SQLite-Datenbank aufbauen | 316 |
| | 15.5.3 Formulierung von Anfragen (Queries) mit SQL | 317 |
| | 15.5.4 Datenbankanfragen in Python-Programmen | 318 |
| | 15.5.5 SQL-Anweisungen mit variablen Teilen | 320 |
| 15.6 | Projekt: Zitatesammlung | 320 |
| | 15.6.1 ER-Diagramm | 321 |
| | 15.6.2 Tabellen (Beispiel) | 321 |
| | 15.6.3 Administration der Zitatesammlung | 322 |
| | 15.6.4 Nach Zitaten suchen | 324 |
| 15.7 | Rückblick | 327 |
| 15.8 | Übungen | 328 |
| 15.9 | Lösungen zu den Fragen | 329 |
| 16 | Wissenschaftliche Projekte | 331 |
| 16.1 | NumPy – Rechnen mit Arrays | 331 |
| | 16.1.1 Arrays | 331 |
| | 16.1.2 Indizieren | 336 |
| | 16.1.3 Slicing | 337 |
| | 16.1.4 Arrays verändern | 338 |
| | 16.1.5 Arithmetische Operationen | 340 |
| | 16.1.6 Funktionen, die elementweise ausgeführt werden | 341 |
| | 16.1.7 Matrizenmultiplikation mit dot() | 341 |
| | 16.1.8 Array-Funktionen und Achsen | 342 |

| | | |
|-----------|---|------------|
| 16.2 | Datenvisualisierung mit matplotlib | 343 |
| 16.2.1 | Liniendiagramme | 344 |
| 16.2.2 | Mehrere Linien in einem Diagramm | 346 |
| 16.2.3 | Histogramme | 347 |
| 16.2.4 | Projekt: Würfeln | 348 |
| 16.2.5 | Heatmaps | 349 |
| 16.3 | Projekt: Bewegungsprofil | 350 |
| 16.4 | Google Colaboratory – Colab | 354 |
| 16.4.1 | Ein Colab-Notebook erzeugen | 354 |
| 16.4.2 | Text-Zellen | 356 |
| 16.4.3 | Bilder einfügen | 358 |
| 16.4.4 | Notebooks speichern und öffnen | 359 |
| 16.5 | Projekt: Füchse und Hasen – Simulation eines Räuber-Beute- Systems | 360 |
| 16.5.1 | Notebooks teilen | 363 |
| 16.6 | Rückblick | 364 |
| 16.7 | Übungen | 365 |
| 16.8 | Lösungen zu den Fragen | 367 |
| 17 | Dynamische Webseiten: CGI und WSGI | 369 |
| 17.1 | Dynamische Webseiten mit CGI | 370 |
| 17.1.1 | Projekt: Wie spät ist es? | 371 |
| 17.1.2 | Die Ausgabe eines CGI-Skripts | 372 |
| 17.1.3 | Wie ist ein CGI-Skript aufgebaut? | 372 |
| 17.1.4 | CGI-Skripte unter Windows | 373 |
| 17.1.5 | Aufruf mit dem Webbrowser | 373 |
| 17.1.6 | Ein HTTP-Server | 374 |
| 17.1.7 | Zugriff von einem anderen Computer im lokalen Netz | 375 |
| 17.2 | Interaktive Webseiten | 375 |
| 17.2.1 | Eingabekomponenten in einem HTML-Formular | 377 |
| 17.2.2 | Verarbeitung von Eingabedaten mit FieldStorage | 379 |
| 17.3 | Wie verarbeitet man Umlaute? * | 380 |
| 17.4 | Dynamische Webseiten mit WSGI | 382 |
| 17.4.1 | Das Applikationsobjekt | 382 |
| 17.4.2 | Skripte mit eigenem HTTP-Server – das Modul wsgiref ... | 383 |
| 17.5 | Projekt: Wie spät ist es? (II) | 383 |
| 17.6 | Projekt: Umfrage | 386 |
| 17.6.1 | Die HTML-Schablonen | 387 |
| 17.6.2 | Der algorithmische Teil | 388 |

| | | |
|-----------|---|------------|
| 17.7 | Einen Hosting-Dienst nutzen | 390 |
| 17.7.1 | Python Anywhere | 390 |
| 17.7.2 | Das vorgefertigtes WSGI-Programm ausprobieren | 390 |
| 17.7.3 | Projekt: Wie spät ist es? (III) | 393 |
| 17.7.4 | WSGI-Projekte modularisieren | 394 |
| 17.8 | Rückblick | 395 |
| 17.9 | Übungen | 395 |
| 17.10 | Lösung zur Frage: Interaktive Webseite | 397 |
| 18 | Professionelle Software-Entwicklung | 399 |
| 18.1 | Die Laufzeit von Programmen | 399 |
| 18.1.1 | Schnelles Sortieren – Quicksort versus Straight Selection . | 399 |
| 18.1.2 | Performance-Analyse mit dem Profiler | 402 |
| 18.2 | Agile Software-Entwicklung | 403 |
| 18.2.1 | Software Engineering | 403 |
| 18.2.2 | Einige Grundideen der agilen Software-Entwicklung | 404 |
| 18.3 | Projekt: Digitales Notizbuch | 405 |
| 18.3.1 | Stories | 405 |
| 18.3.2 | Erste Iteration | 406 |
| 18.3.3 | Zweite Iterationen | 407 |
| 18.3.4 | Refactoring | 408 |
| 18.3.5 | Neue Stories und Änderbarkeit | 412 |
| 18.4 | Test Driven Development mit doctest | 413 |
| 18.5 | Übung: Ticketbuchung | 415 |
| | Glossar | 417 |
| | Stichwortverzeichnis | 425 |