Rezension

Weißer, Martin. 2022. *Python-Programmierung für Germanist:innen. Ein Lehr- und Arbeitsbuch*. Tübingen: Narr. 224 S., 28,90€, ISBN: 978-3-8233-8456-4.

Besprochen von **Marcella Palladino,** Università degli Studi di Modena e Reggio Emilia, Dipartimento di Studi Linguistici e Culturali, Largo Sant'Eufemia 9, 41121 Modena (Italien),
E-Mail: marcella.palladino@unimore.it

https://doi.org/10.1515/zfal-2026-2010

Weißers Buch besteht aus zehn thematischen Kapiteln, einem Schlusswort (Kap. 11) und einem Appendix (Kap. 12). In der Einleitung (Kap. 1) macht der Autor deutlich, dass sein Band als eine Einführung in die Programmierung mit Python konzipiert ist. Aus diesem Grund werden keine Vorkenntnisse der Leser:innen vorausgesetzt. Python stellt zahlreiche Vorteile für Forscher:innen der Germanistik dar und insbesondere die Möglichkeit, einen einheitlichen methodischen Arbeitsprozess zu verwenden, ohne mehrere zweckgebundene Tools beherrschen zu müssen. Ziel des Buches ist es laut Weißer, Grundbegriffe der Programmierung für die Sprachanalyse anzubieten und damit Analysen zu ermöglichen, die mit bestehenden Programmen nicht realisierbar wären (S. 10). Die Einleitung dient zudem der Einführung in die Darstellungskonventionen des Buches, welche den Zugang zu den folgenden Inhalten erleichtern. Der Autor wendet sich direkt an die Leser:innen und etabliert einen dialogischen Ton, um die Erklärungen zu vermitteln. Darüber hinaus werden praktische Aspekte in diesem Kapitel behandelt: die Installation von Python, eine Einführung in die Dateisysteme, die Illustrierung der Editoren und der IDEs (*Integrated Development Environments*) sowie Anweisungen zur Installation und Einrichtung von *WingIDE Personal*. Weißer präzisiert zusätzlich, wie eine deutsche Tastatur mit Sonderzeichen eingerichtet werden soll. Solche Aspekte werden oft in Lehr- und Arbeitsbüchern vorausgesetzt und daher müssen Leser:innen oft andere Quellen suchen, falls Zweifel entstehen. Eine Stärke dieses Buches liegt in den ausführlichen Erklärungen des Autors, die besonders für Anfänger:innen hilfreich sind. Die Darstellung der Inhalte wird durch Übungen ergänzt, deren Lösungen sich am Ende des jeweiligen Kapitels befinden. Dadurch können Leser:innen selbständig üben und gleichzeitig ein Feedback zu ihren Aufgaben bekommen.

Kapitel 2 – *Grundlagen der Programmierung I* – ist mit Kapitel 3 – *Grundlagen der Programmierung II* – verbunden. Der Autor führt hier in Anweisungen, Funktionen und Variablen ein und erläutert damit verbundene Begriffe wie bspw. Argumente. Auch einfache Datentypen werden eingeführt. Weißer verweist auf die Relevanz dieser Begriffe für die folgenden Kapitel, um deutlich zu machen, dass die

hier vermittelten Grundbegriffe eine wesentliche Grundlage für die folgenden Programmierkonzepte darstellen. Besonders relevant ist der Unterabschnitt über die Erstellung von Skripten und Programmen, die über einen Editor oder eine IDE erfolgen kann (S. 33). In diesem Zusammenhang betont der Autor die Bedeutung der Kommentierung des Codes. Dieser Aspekt ist insbesondere in kooperativen Forschungsprojekten von Relevanz, da Kommentare es erlauben, die Programmcodes nachzuvollziehen. Außerdem sind Kommentare auch für die eigene Arbeit hilfreich, weil sie den Denkprozess dokumentieren und den späteren Rückgriff auf den eigenen Code erleichtern. In Kapitel 2 widmet Weißer zusätzlich den Lösungen der Übungsaufgaben zahlreiche Seiten, in denen er typische Fehler und deren Korrekturen aufzeigt. Kapitel 3 folgt der Struktur der vorherigen Teile und führt in zusammengesetzte Datentypen, Interaktionen und die unterschiedlichen Sorten von Schleifen ein. Auch hier arbeitet Weißer mit Tabellen, die Kategorien wie z.B. Datentypen effizient zusammenfassen und sie schematisch darstellen.

Kapitel 4 ist den Grundlagen der Zeichenkettenverarbeitung gewidmet. Der Autor erklärt, wie Zeichenketten interpretiert, bereinigt, extrahiert und formatiert werden können. Die für diesen Zweck relevanten Funktionen werden zusammen mit Sequenzen und Listen ausführlich dargestellt.

In Kapitel 5 stehen gespeicherte Daten im Mittelpunkt. Der Fokus liegt auf die Datentypen, die für Germanist:innen relevant sein können, d.h. vor allem Textdateien. Dateien werden in Python als Objekte verstanden, und Weißer illustriert die Aktivitäten, die mit ihnen möglich sind, z.B. Inhalte auslesen oder in Dateien schreiben. Zudem wird es gezeigt, wie man mit Verzeichnissen und Pfaden arbeiten kann.

Kapitel 6 fokussiert sich auf die Erkennung und Bearbeitung von Sprachmustern. Im Vordergrund stehen reguläre Ausdrücke (abgekürzt *Regexes*), die es ermöglichen, Daten zu verarbeiten und komplexere Muster zu definieren sowie zu suchen (S. 93). Wie bereits erwähnt, kann Python die Arbeit mit zahlreichen unterschiedlichen Programmen ersetzen, und genau in diesem Zusammenhang gewinnen reguläre Ausdrücke an Relevanz: sie bieten vielfältige Funktionen, die dabei helfen können, Sprachanalysen durchzuführen und komplexe Muster zu dekonstruieren. Ein Beispiel dafür ist *search* (Muster, Kette) für die Identifikation eines (ersten) Auftretens eines Musters in einer Zeichenkette (S. 93). Hinsichtlich korpuslinguistischer Verfahren bedeutet dies, dass die Belege eines Lemmas direkt mittels regulärer Ausdrücke extrahiert bzw. berechnet werden können, ohne auf externe Programme zurückgreifen zu müssen, die häufig spezifische Dateiformate erfordern. Auch das Treffer-Objekt erweist sich in diesem Zusammenhang als besonders hilfreich (S. 95). Weißer vertieft Zeichenklassen, Fehlerbehandlung bei Regexes sowie Gruppierungen der Regexes. Das Kapitel endet mit Kompilierungsflags, mit denen Eigenschaften der Regexes modifiziert werden können.

Kapitel 7 ist in mehrere thematische Unterabschnitte gegliedert. Zuerst werden *Dictionaries* eingeführt, die Schlüssel und Werte sammeln. Besonders wichtig sind Module, die häufig verwendete Funktionen und Klassen enthalten. Weißer fokussiert sich lediglich auf einfache Module – einfache Python-Dateien, die wie interne Module importiert werden können – und zeigt, wie sie getestet und installiert werden. Komplexere Module, die oft als Pakete installiert werden, werden hingegen ausgeschlossen. Diese Entscheidung entspricht dem Einführungscharakter des Buches. Allerdings könnten auch komplexere Module für die Germanist:innen interessant sein, insbesondere wenn sie spezifische Analysen erleichtern. Daher könnte eine sinnvolle Erweiterung dieses Bandes auf diese Aspekte eingehen. Weißer vertieft danach Klassen und Objekte, bevor das Kapitel mit den Lösungen der Aufgaben abgeschlossen wird.

Kapitel 8 ist aus linguistischer Perspektive besonders relevant. Im Fokus stehen Wortlisten, Frequenzlisten für Wörter, N-Gramme und Sortieroptionen für diese Listen. Diese sprachlichen Phänomene und deren Quantifizierung spielen für Sprachanalysen eine grundlegende Rolle. Weißer erläutert, wie Wortlisten generiert werden können und welche Sortieroptionen für listenartige Objekte bestehen. Zudem zeigt er, wie Frequenzlisten generiert werden, und führt den in Programmiersprachen üblichen Begriff der Lambda-Funktionen ein. Diese Funktionen benötigen keine explizite Benennung, sondern werden ad hoc im Code definiert und eingesetzt (S. 137). Ein weiteres Thema vor den Lösungen der Aufgaben ist die relative Frequenz, die Vergleiche zwischen verschiedenen Korpora oder Dateien ermöglicht (im Gegensatz zu rohen Frequenzen). Allerdings wird im Kapitel nicht im Detail erläutert, wie N-Gramme erzeugt werden, was im Hinblick auf ihre grundlegende Bedeutung für linguistische Analysen als erklärungsbedürftig angesehen werden kann.

Kapitel 9 befasst sich mit den grafischen Benutzeroberflächen (GUIs). Ihre Funktion besteht darin, eine vereinfachte Interaktion zwischen Daten und Benutzer:innen eines Programms zu ermöglichen. Die Bibliothek *PyQt* ist ein zentrales Werkzeug, das Steuerelemente wie Fenster oder graphische Objekte enthält. Weißer betont, dass die Benutzer:innen die zu erfüllenden Aufgaben zuerst definieren müssen, um die geeigneten GUI-Elemente auszuwählen. Im Mittelpunkt des Kapitels stehen Layouts, die die Gruppierung und Anordnung der Steuerelemente erlauben. Der letzte Unterabschnitt des Kapitels widmet sich dem Umgang mit Dateien und Verzeichnissen in *PyQt*. Weißer greift drei unterschiedliche Methoden auf und zeigt syntaktische Beispiele, die im Code verwendet werden können. Die Darstellung praktischer Beispiele erscheint didaktisch besonders sinnvoll, da die Komplexität der Themen im Buch progressiv zunimmt und die Leser:innen durch die Beispiele eine Unterstützung erhalten. M.E. fehlt den Kapiteln eine Zusammenfassung am Ende, in der eine Übersicht der wichtigsten behandelten Aspekte und deren Rele-

vanz für die folgenden Kapitel dargestellt wird. Solche Zusammenfassungen wären nicht nur geeignet, um zentrale Inhalte zu betonen, sondern würden auch das Erinnern grundlegender Konzepte erleichtern, was vor allem hilfreich ist, falls die Lektüre des Buches in mehreren Etappen erfolgt.

Das letzte thematische Kapitel (Kap. 10) befasst sich mit Webdaten und Annotationen. Der Autor zeigt, wie Dateien aus dem Web verwendet werden können, und im Vordergrund stehen die Formate HTML und XML. HTML-Seiten können in Python heruntergeladen und bearbeitet werden. Das für die Illustration in diesem Kapitel ausgewählte Modul ist *Beautiful Soup*, welches erlaubt, Elemente und Texte in Webseiten zu finden sowie zu manipulieren bzw. durch sie zu navigieren (S. 176). Weißer geht auf das XML-Format ein und erklärt, wie Texte in XML konvertiert werden und welche Aspekte (z.B. der strukturelle und inhaltliche Aufbau des Textes) dabei zu berücksichtigen sind. Obwohl im Kapiteltitel explizit auch Annotationen genannt werden, werden sie nur kurz in Bezug auf HTML und XML erwähnt. Es wird hervorgehoben, dass XML für linguistische Annotierungen besser geeignet ist und dass Tags möglichst selbsterklärend gestaltet werden sollten. Es fehlt ein Abschnitt, der sich systematisch mit den methodischen und praktischen Herausforderungen linguistischer Annotationen auseinandersetzt, die für die linguistische Forschung besonders relevant sein könnten.

Kapitel 11 ist das Schlusswort, in dem Weißer betont, dass das Kennen der wichtigsten Grundlagen der Programmierung für die Entwicklung einer Programmierkompetenz nicht reicht: Diese entsteht vielmehr durch regelmäßige Übung und praktische Erfahrung. Im Zusammenhang damit ist die Entscheidung, zahlreiche Aufgaben und deren Lösungen in das Buch zu integrieren, konsequent und didaktisch konsistent. Sie geben den Leser:innen die Möglichkeit, die eingeführten Begriffe praktisch anzuwenden und sich schrittweise in Python einzuarbeiten. Im Appendix (Kap. 12) sind die vollständigen Codes der vorgestellten Python-Programme zu finden, die eine wichtige Ressource für die Leser:innen und ihre weiteren Projekte darstellen.

Weißers Buch hat die Stärke, eine schrittweise und nachvollziehbare Einführung in die Verwendung von Python für Germanist:innen zu bieten, welche sowohl für Anfänger:innen als auch für fortgeschrittene Lerner:innen, die bestimmten Aspekte wiederholen möchten, geeignet ist. Es richtet sich besonders an Germanist:innen, die Python als methodisches Instrument verwenden möchten, um Arbeitsschritte zu vereinfachen bzw. zu beschleunigen. In der heutigen linguistischen Forschung kann eine grundlegende Programmierkenntnis wesentlich dazu beitragen, Dateien effizient zu sammeln und Analysen durchzuführen, die sonst sehr zeitaufwendig oder kaum möglich wären. Besonders Kapitel 10 zeigt deutlich, welche Relevanz Webdaten für die linguistische Forschung besitzen und wie sie mit Python für eigene Analysen verwendet werden können.

Das Buch positioniert sich als klar strukturiertes und didaktisch reflektiertes Grundlagenwerk im Bereich der Digital Humanities, ließe sich aber um weitere Aspekte ergänzen. Beispielsweise wären Erweiterungen durch Pakete und Programme zur Arbeit mit gesprochener Sprache sinnvoll, denn linguistische und germanistische Forschung beschränkt sich nicht allein auf schriftliche Texte, sondern untersucht auch mündliche Kommunikationsformen. Gleichzeitig ist die Beschränkung auf die schriftliche Sprache aus didaktischer Perspektive nachvollziehbar, um die Anfänger:innen nicht zu überfordern. Allerdings wäre eine Erweiterung der Buchstruktur um Methoden zur Analyse gesprochener Sprache wünschenswert, um einen vollständigeren Möglichkeitsrahmen zu vermitteln.

Im Verlauf der Kapitel stellt man sich die Frage, warum das Buch gezielt an Germanist:innen und nicht allgemein an Linguist:innen gerichtet ist. Abgesehen von den expliziten Hinweisen des Autors ist der Inhalt des Buches nicht nur auf die spezifischen Bedürfnisse der Germanistik beschränkt. Vielmehr fungiert das Buch als geeignete Einführung für Linguist:innen allgemein, die grundlegende Programmierkenntnisse mit Python anhand praktischer Übungen erwerben möchten. Weiterhin ist Python nur eine der für die Linguistik relevanten Programmiersprachen, weswegen das Buch als Teil eines größeren Themenkomplexes verstanden werden könnte, der die Anwendung verschiedener Programmiersprachen und ihre Relevanz für linguistische Analysen thematisiert.