Lernaufgaben digital dokumentieren, sammeln und veröffentlichen

Robert Ringel[1]

DOI: 10.55310/jfhead.74

Abstract

Sammlungen von Lernaufgaben sind eine wichtige Grundlage für die Anwendung von Methodiken des aufgabenbasierten Lernens. Die Konzeption von Aufgabensammlungen muss dabei sowohl die methodischen Aspekte der Lernaufgaben als auch die inhaltliche Organisation der Aufgaben berücksichtigen. Soll die Aufgabensammlung in digitaler Form publiziert werden, ist zudem ein leicht zugängliches und servicefreundliches digitales Medium auszuwählen. In diesem Beitrag wird anhand eines konkreten Pools von Aufgaben zum Programmierenlernen gezeigt, wie diese Anforderungen praktisch erfüllt werden können. Dafür wird ein spezifischer Satz von Meta-Daten anhand vorliegender Fachliteratur zusammengestellt und an einem konkreten Beispiel illustriert. Der dargestellte Aufgabenpool wurde mit Hilfe von Markdown in GitHub publiziert und umfasst mehr als 100 Lernaufgaben für die Python-Programmierung. Der Beitrag regt dazu an, die Konzeption auch auf andere naturwissenschaftlich-technische Fachgebiete zu übertragen.

Keywords

Aufgabensammlung; Programmieraufgaben; Lernaufgaben; Meta-Daten; OER

1 Robert Ringel
Wissenschaftlicher Mitarbeiter, HTW Dresden, Fakultät Informatik/Mathematik
robert.ringel@htw-dresden.de

1. Einleitung

Lernaufgaben sind das zentrale Element von Lernprozessen. Lehrpersonen im Bereich der Hochschullehre orientieren sich bei der Aufgabengestaltung häufig an eigenen praktischen Erfahrungen. Dabei entsteht für jede Lehrveranstaltung meist ein spezifischer Satz an Aufgaben. Das Ziel dieses Beitrages besteht darin aufzuzeigen, wie Lernaufgaben eines Teilbereiches der Informatik gesammelt werden können, um sie nachhaltig und effizient in digitaler Form als öffentliche Bildungsressource (Open Educational Ressource, OER) zu publizieren. Dieses Paper zeigt am Beispiel eines realen Aufgabenpools aus dem Bereich des Programmierenlernens, wie die systematische Aufgabengestaltung und Aufgabensammlung erfolgen kann. Der Beitrag gibt eine praktische Orientierung für die Gestaltung von Aufgabensammlungen. Zunächst werden grundlegende Aufgabenklassen komplexer Lernprozesse dargestellt. Daraus folgt die Ableitung eines Satzes elementarer Metainformationen zur Klassifikation von Lernaufgaben, welcher die Voraussetzung für die Organisation und Kuratierung eines Aufgabenpools bildet. Anhand eines GitHub-Repositories mit Aufgaben der Python-Programmierung wird die Gestaltung und Nutzung eines Aufgabenpools illustriert. Dabei wird deutlich, dass die gezeigte konzeptionelle Gestaltung der Aufgabensammlung Anregungen für die Übertragung in andere Fachbereiche geben kann.

2. Elementare Aufgabentypen zur gezielten Gestaltung von Aufgabenfolgen

Merriënboer und Kirschner (2018) beschreiben in „Ten steps to complex learning“ das Vorgehen der Aufgabengestaltung für komplexe Lernprozesse beruflicher und akademischer Ausbildung. Im Kern der Methodik des „four components instructional design“ (4C/ID) stehen Folgen von Lernaufgaben.

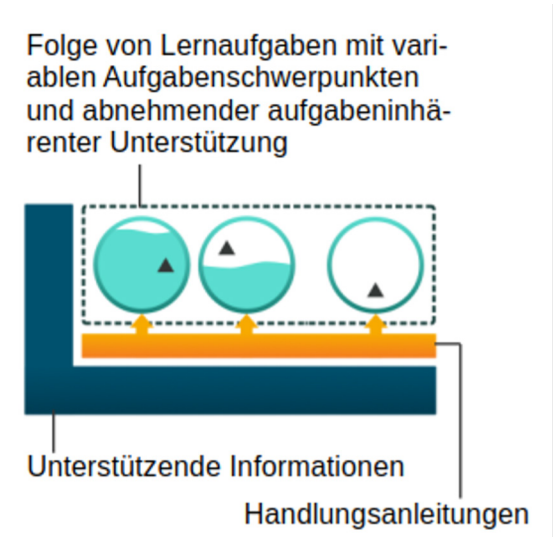

Abb. 1: Schematische Darstellung des Gestaltungsprinzips von Aufgabenfolgen gemäß 4C/ID (2025)

Diese Aufgabenfolgen beginnen mit solchen Aufgaben, die einen hohen Grad an inhärenter Unterstützung enthalten. Der Grad der Unterstützung nimmt von Aufgabe zu Aufgabe systematisch ab, woraus ein steigendes Anforderungsniveau zur Lösung der Aufgaben resultiert. Die Aufgabenfolge ist insgesamt so konzipiert, das fachliche Lernziele variieren, um einen induktiven Lernprozess zu fördern. Abbildung 1 zeigt dieses Schema von Aufgabenfolgen nach 4C/ID.

Für die Ausgestaltung derartiger Aufgabenfolgen schlagen Merriënboer und Kirschner (2018, S. 72) sechs grundlegende Aufgabentypen vor, die in den folgenden Abschnitten beschrieben werden. Es erfolgt eine kurze Darstellung der Spezifik des Aufgabentyps und der Lernhandlungen. Lernhandlungen sind jene aufgabenbezogenen Tätigkeiten, die im Prozess der Aufgabenbearbeitung den Kompetenzaufbau fördern.

Beschreibung der Aufgabentypen

Das ausgearbeitete Beispiel (worked-out example) enthält aufgrund der gegebenen Aufgabenlösung den höchstmöglichen aufgabeninhärenten Unterstützungsgrad. Die Aufgabenstellung besteht zumeist darin, den Lösungsweg nachzuvollziehen, zu dokumentieren, zu kommentieren oder zu beschreiben. Die Lernhandlungen bestehen primär im Lesen und Verstehen der vorgelegten ausgearbeiteten Lösung.

Die Aufgabenumkehrung (reverse task) besteht ebenfalls aus der Darstellung einer vollständigen Aufgabenlösung. Die Aufgabenstellung verlangt die Formulierung des zugehörigen Aufgabentextes durch die Lernenden. Damit einher gehen Lernhandlungen des Lesens und Verstehens der vorliegenden Aufgabenlösung und der schriftlichen Formulierung eines Arbeitsauftrages, der diese Lösung erzeugt. So sind die Lernenden aufgefordert, ihr Verständnis zu erarbeiten und zu dokumentieren. Durch das Vorliegen der Aufgabenlösung bietet dieser Aufgabentyp ebenfalls eine hohe aufgabeninhärente Unterstützung.

Die Nachahmungsaufgabe (imitation task) verlangt die Nachahmung einer vorgelegten oder bekannten Aufgabenlösung für einen gleichartigen Fall, den die Lernenden selbst wählen. Die Lernhandlungen bestehen im Erkennen der gegebenen Ausgangssituation, des erreichten Zielzustandes und des realisierten Lösungsprozesses der bekannten Aufgabenlösung. Hinzu kommt das Ziehen von Analogieschlüssen für die Übertragung und Realisierung des gewählten analogen Falles. Damit bietet auch dieser Aufgabentyp eine gute fachliche Unterstützung, die mit den Anforderungen einer analogen Transferleistung kombiniert wird.

Die Aufgabe mit freier Zielwahl (non-specific goal) beschreibt eine Ausgangssituation, beispielsweise als Satz gegebener Parameter oder als ein physikalischer Zusammenhang, wofür eine selbst gewählte fachliche Anwendung zu realisieren ist. Lernhandlungen umfassen dabei die fachlich korrekte Einordnung der Ausgangssituation und den kreativen, anwendenden Transfer für eine ausgewählte Problemlösung. Somit bietet dieser Aufgabentyp nur eine anfängliche inhaltliche Unterstützung, bei gleichzeitig gesteigerten Anforderungen im fachlichen Verständnis und Transfer.

Eine Vervollständigungsaufgabe (completion task) enthält die Darstellung der Ausgangs- und der Zielsituation (gegeben und gesucht) sowie einen Teil der Problemlösung. Die Lernhandlungen verlangen die Vervollständigung des vorliegenden Lösungsansatzes. Dafür ist der Lösungsansatz zu lesen, nachzuvollziehen, konzeptionell zu verstehen sowie durch das Auswählen und das Realisieren fehlender Lösungselemente zu ergänzen bzw. zu Ende zu führen. Diese Aufgabenklasse bietet eine anfängliche aufgabeninhärente Unterstützung bei gleichzeitigen erheblichen fachlichen Anforderungen zum Erkennen und Vervollständigen des vorliegenden Lösungskonzeptes.

Die gewöhnliche Aufgabenstellung (conventional task) besteht aus einer gegebenen Ausgangssituation, die in einen gesuchten Zielzustand zu überführen ist. Sie erfordert Lernhandlungen zum Lesen und Verstehen der Aufgabenstellung, um den Ausgangs- und den Zielzustand zu erkennen. Auf dieser Grundlage ist eine Problemlösungsstrategie auszuwählen und zu realisieren. Idealerweise sollte die erbrachte Problemlösung rückschauend bewertet werden. Die gewöhnliche Aufgabenstellung stellt nur minimale aufgabeninhärente Unterstützung bereit und erfordert – im Vergleich zu den hier dargestellten Aufgabentypen – die größten fachlichen Anstrengungen.

Tabelle 1 zeigt diese sechs Aufgabentypen in einer Gegenüberstellung. Bei der Betrachtung der Aufgabentypen fällt auf, dass mehrere Typen vollständige oder teilweise vorliegende Aufgabenlösungen enthalten, die von den Lernenden zu lesen sind. Nur bei der gewöhnlichen Aufgabenstellung wird die vollständige Entwicklung einer spezifischen gesuchten Problemlösung verlangt. Die wissenschaftliche Grundlage für diese Gestaltung von Lernaufgaben liefert Merriënboer (1990) in seinem Aufsatz „Strategies for programming instruction in high school“. Hier zeigt er am Beispiel des Programmierenlernens, dass es für den Beginn eines Lernprozesses wichtig ist, mehrere Lösungsbeispiele zu lesen und zu verstehen, bevor Problemlösungen selbst zu erarbeiten sind.

Die Beschreibung der grundlegenden Aufgabenklassen nach 4C/ID bildet den konzeptionellen Ausgangspunkt für die Gestaltung von Aufgabenfolgen. Die Kenntnis der Aufgabenklassen gibt Anregungen, wie Lernziele und Lernhandlungen unter Verwendung aufgabeninhärenter Unterstützung gezielt in vielfältigen und variablen Aufgabenstellungen kombiniert werden können. Sie gibt eine methodische Basis für

die Gestaltung von Aufgabenfolgen und die Verwaltung von Aufgabensammlungen.

Aufgabentyp	gegeben	gesucht	Lösung	übergeordnetes Lernziel
ausgearbeitetes Beispiel	x	x	x	Verständnis der Lösung erarbeiten und dokumentieren
Aufgabenumkehrung	angeben	x	x	Verständnis der Lösung erarbeiten und durch Formulierung der zugehörigen Aufgabenstellung dokumentieren
Nachahmungsaufgabe	x/analog	x/analog	x/analog	Ausgangspunkt, Ziel und Lösung ausgehend von einer gegebenen Aufgabe für eine selbst gewählte analoge Aufgabe transferieren
Aufgabe mit freier Zielwahl	x	definieren	finden	für gegebenen Ausgangspunkt ist ein Ziel zu definieren und die zugehörige Lösung anzugeben (Transferleistung mit selbst definiertem Anforderungsniveau)
Vervollständigungsaufgabe	x	definieren	finden	Zielstellung, Lösungsdetails und ggf. Lösungsstrategie sowie deren Umsetzung erkennen und vervollständigen
gewöhnliche Aufgabenstellung	x	x	finden	zu lösendes Problem erkennen, Strategie finden und realisieren sowie prüfen; Umfang des gestellten Problems bestimmt die Aufgabenanforderung

Tab. 1: Übersicht zu Aufgabentypen nach Merriënboer und Kirschner (2018, S. 72)

3. Gestaltung und Pflege von Aufgabensammlungen

Bei der Gestaltung und Pflege von Aufgabensammlungen besteht die grundlegende Notwendigkeit zur Differenzierung von Lernaufgaben und Testaufgaben. Lernaufgaben haben das Ziel, dass die Lernenden bei der Aufgabenbearbeitung Lernhandlungen vollziehen, die einen Kompetenzaufbau ermöglichen. Demgegenüber besteht der Zweck von Testaufgaben in der Diagnose erreichter Kompetenzen gemessen an den bestehenden Qualifikationszielen. Im Fokus dieses Beitrags stehen die Lernaufgaben mit den jeweiligen Lernzielen und Lernhandlungen.

Für die Gestaltung und Pflege von Aufgabensammlungen bedarf es eines Satzes an Merkmalen oder Kriterien zur Ordnung bzw. Klassifikation der Menge an Aufgaben. Einen Ansatz liefert Astleitner mit dem kategorialen Modell von Aufgabenmerkmalen (Astleitner, 2006, S. 26). Er benennt insgesamt 14 verschiedene Aufgabenmerkmale. Darin enthalten sind Informationen wie Aufgabenansatz, -komplexität und -zielniveau sowie die Lernunterstützung für die Aufgabenbearbeitung. Ergänzend wären denkbar: Aufgabentyp (z.B. gemäß 4C/ID), Lernbereich und fachlicher Themenbereich einer Aufgabe. Einen weiteren umfassenden Katalog an didaktischen Metadaten, die auch für Lernaufgaben nutzbar sind, liefern Oellers und Rörtgen (2024) im „Kompendium: Didaktische Metadaten“. Die Schwierigkeit besteht darin, einen solchen Satz Metadaten zu definieren, der sowohl aussagekräftig als auch umfassend genug ist, um das Ordnen und Auffinden von Lernaufgaben in einer Aufgabensammlung zu ermöglichen. Dabei ist insbesondere die fachliche Spezifik der Aufgabensammlung zu beachten.

Die Lernziel-Taxonomie nach Bloom (1956) ist ein generisches, fachunabhängiges Schema, um Lernziele darzustellen. Sie eignet sich zum Beispiel, um die fachliche Abdeckung von Prüfungen gegenüber den Lernzielen eines Moduls zu dokumentieren. Die von Krathwohl (2002) vorgeschlagene Verwendung von Verben benutzt vorranging charakterisierende, universelle Verben, was dem Anliegen der Taxonomie Rechnung trägt. Für die Dokumentation einzelner Lernaufgaben – insbesondere für Problemlöseprozesse im naturwissenschaftlich-technischen Bereich

– erscheint es sinnvoll, spezifische Lernhandlungen zu benennen. Diese aufgabenspezifischen Lernhandlungen werden bei der Bearbeitung der Lernaufgaben ausgeführt. Die wiederholte Ausführung der Lernhandlungen in verschiedenen Aufgabenkontexten ermöglicht den Kompetenzaufbau, welcher sich in der Erreichung der Lernziele einer Lehrveranstaltung zeigt. Ausgehend von diesen Überlegungen wird hier vorgeschlagen, die folgenden Merkmale mit didaktischer Bedeutung für die Aufgabendokumentation zu verwenden: Das spezifische Lernziel der Aufgabe, die notwendigen Vorkenntnisse zur Aufgabenbearbeitung sowie die Lernhandlungen. Zur Orientierung und Suche innerhalb einer Aufgabensammlung sind folgende Kriterien hilfreich: Lernbereich, Aufgabenkomplexität und ggf. fachlicher Themenbereich einer Aufgabe.

Die Aufgabenkomplexität dient als Indikator für das Anforderungsniveau der Aufgabenstellung. Sie kann als Hilfe bei der Aufgabenauswahl dienen. Chen, Paas und Sweller (2023, S. 63) zeigen eine Übersicht möglicher Indikatoren der Aufgabenkomplexität. Daraus wird deutlich, dass sich die Indikatoren der Komplexität bezüglich verschiedener Merkmale einteilen lassen und das dabei Abhängigkeiten zu anderen Aufgabenmerkmalen und Merkmalen der lernenden Person entstehen können. Sowohl Merriënboer und Kirschner (2018, S. 16) als auch Chen, Paas und Sweller (2023) stimmen grundsätzlich überein, dass Komplexität aus potentiellen oder realen Interaktionen verschiedener Elemente im Lösungsprozess einer Aufgabenstellung erwächst. Darauf aufbauend und mit dem Ziel, ein praktikables und objektives Komplexitätsmaß zu definieren, wird folgendes vorgeschlagen: Für die Komplexität einer Lernaufgabe wird primär die Anzahl der enthaltenen Lernbereiche betrachtet. Eine niedrige Aufgabenkomplexität liegt bei der Interaktion von 1-2 Lernbereichen vor. 3-4 interagierende Lernbereiche repräsentieren eine mittlere Komplexität. Eine höhere Anzahl Lernbereiche legt eine hohe Aufgabenkomplexität nahe. Der Anwendungskontext kommt als ein sekundäres Komplexitätsmerkmal hinzu. Erfordert der Anwendungskontext der Aufgabe spezifisches Zusatzwissen, erhöht sich Aufgabenkomplexität um eine Stufe. So kann die Komplexität auf einer 4-stufigen Skala unabhängig von subjektiven Eigenschaften der lernenden Person abgebildet werden. Tabelle 2 zeigt die resultierende Modellierung der Komplexitätsskala.

Aus den genannten Merkmalen zur Charakterisierung von Lernaufgaben lässt sich die Informationsstruktur einer Aufgabensammlung ableiten, wie sie in Tabelle 3 dargestellt ist. Darin werden die einzelnen Merkmale mit Beispielangaben einer konkreten Programmieraufgabe illustriert. Diese Informationen gilt es systematisch mit Hilfe einer Dokumentvorlage zu erfassen, um die Lernaufgaben in den Aufgabenpool zu integrieren. Daraus werden nachfolgend die Aufgabendarstellung sowie zusammenfassende Übersichtsdarstellungen für Suche und Orientierung generiert.

Komplexität	Anwendungskontext basiert auf Allgemeinwissen	Anwendungskontext erfordert Zusatzwissen
1-niedrig	1-2 Lernbereiche	–
2-normal	3-4 Lernbereiche	1-2 Lernbereiche
3-hoch	> 4 Lernbereiche	3-4 Lernbereiche
4-komplex	–	> 4 Lernbereiche

Tab. 2: Modellierung der Komplexitätsskala aus Anzahl der Lernbereiche und Anwendungskontext

Merkmal	Inhalt	Beispielangaben für eine Programmieraufgabe
Aufgabentitel	aussagekräftiger Titel mit Angabe einer thematischen Information	Berechnung von Flaschenpfand
Lernbereich	Lernbereich innerhalb eines Kurses oder einer Lehrveranstaltung	Eingabe-Verarbeitung-Ausgabe
Lernziel	Angabe zum angestrebten Aufbau einer spezifischen Teilkompetenz	Verständnis für Variablen und Berechnungen
Vorkenntnisse	notwendige Kenntnisse für die Bearbeitung der Aufgabe	Wertezuweisungen, numerische Typumwandlung bei Tastatureingabe, print-Anweisung
Aufgabentext	exakte Formulierung der Aufgabenstellung	Das folgende Programm berechnet den Pfand bei der Flaschenrückgabe. Lesen Sie den Code und führen Sie das Programm aus. Erweitern Sie den Code, so dass auch ein Dosenpfand (Vergütung 0,25 EUR/Dose) in die Berechnung einbezogen wird.
Aufgabentyp	Angabe des Aufgabentyps gemäß eines Klassifikationssystems	Vervollständigungsaufgabe
Lösungsbeispiel	vollständiges Lösungsbeispiel	Python-Programm
Lösungshilfen, unterstützende Informationen	Lehrbuchseiten oder Link auf Webseite	www.python-kurs.eu/python3_operatoren.php
Lernhandlungen	Nennung wesentlicher Handlungen oder Lösungsschritte der Aufgabenbearbeitung	Lesen und Ausführen des Programms; Codeverständnis erarbeiten; Code gem. Aufgabe erweitern; Ausführen und Testen des Programms
Komplexität	Angabe auf einer definierten Skala	1 - niedrig (Lernbereich: Eingabe-Verarbeitung-Ausgabe)

Tab. 3: Zusammenstellung von Merkmalen zur Klassifikation von Lernaufgaben

Als Dateiformat für die Erfassung und Darstellung der Informationen kann Markdown empfohlen werden. Es ist eine gut lesbare und sehr einfach gestaltete Auszeichnungssprache (Liepins, 2024), die eine Integration von Weblinks, Grafiken und Formeldarstellungen ermöglicht. Markdown-Dateien können sehr gut von Content-Managementsystemen und Dateiverzeichnissen verwaltet werden.

4. Beispiel für eine digitale Sammlung von Programmieraufgaben

Die beschriebenen Konzepte zur Gestaltung von Aufgabensammlungen sollen nun am konkreten Beispiel eines englischsprachigen Aufgabenpools aus dem Bereich der Informatik illustriert werden. Das Beispiel bezieht sich auf Lernaufgaben zum Programmierenlernen, die in GitHub unter folgender URL publiziert sind: https://github.com/RobertRingel/LearningTasks4Programming.

GitHub ist in der Informatik als akzeptiertes Werkzeug zur Verwaltung von Programmquelltexten und Software-Projekten etabliert. Es bietet neben der Verwaltung von Programmversionen eine leistungsfähige Unterstützung für die Arbeit unterschiedlicher Personen an gemeinsamen Projekten unabhängig von Standorten, Organisationen und verwendeten Betriebssystemen. Damit ist GitHub geeignet, den kostenlosen öffentlichen Zugang zur Aufgabensammlung zu gewährleisten und wichtige Werkzeuge zur Kuratierung der Aufgabensammlung bereitzustellen, ohne dass eine spezifische IT-Infrastruktur benötigt wird.

Die Aufgabensammlung besteht aus Markdown-Dateien, die ausgehend von einem Template-Dokument die Aufgabeninhalte enthalten. Markdown ist eine Auszeichnungssprache mit sehr wenigen und einfachen Elementen zur Textgestaltung. Dazu gehören neben Überschriften, Aufzählungen und Elementen der Texthervorhebung auch Möglichkeiten zur Integration von Weblinks und grafischen Darstellungen sowie Programmquelltexten oder Formeln. Markdown-Dateien lassen sich mit einfachen Texteditoren schreiben und zum Beispiel im Webbrowser anzeigen. Das Template-Dokument als Vorlage der Aufgabenpräsentation enthält neben dem Aufgabentext und der Beispiellösung alle Metainformation zur Beschreibung der Aufgabe. Abbildung 2 zeigt das

Template-Dokument in Markdown-Syntax. Zur Dokumentation einer Lernaufgabe sind die aufgabenspezifischen Inhalte in diese Datei einzutragen, bevor die Datei unter einem spezifischen Dateinamen gespeichert wird.

Nach diesem Ansatz wurde die Aufgabensammlung zur Python-Programmierung erstellt und im genannten Git-Hub-Repository unter OER-Lizenz publiziert. Sie enthält mit Stand April 2025 109 Lernaufgaben, die in sechs Themenbereiche eingeordnet sind. Für jeden Themenbereich liegt eine Tabelle als Übersicht aller verfügbaren Aufgaben vor. Abbildung 3 zeigt die Tabelle für den Themenbereich „Steuerung des Programmablaufs: Wiederholungen“. Neben dem Namen und der Komplexität sind darin auch das Lernziel und der Aufgabentyp aufgeführt. Durch einen Mausklick auf den Namen der Aufgabe erfolgt die Weiterleitung zur Aufgabendarstellung selbst. Abbildung 4 zeigt die Darstellung für das Beispiel einer Aufgabendokumentation in diesem Themenbereich und veranschaulicht eine Aufgabenumkehrung (reverse task). Sie enthält ein vollständiges Python-Programm und den Arbeitsauftrag zum Lesen, Analysieren und Ausführen des Programms, um seinen Zweck zu erkennen. Dabei ist der Text des Programmierauftrages, der das vorgegebene Programm beschreibt, schriftlich zu formulieren. Im Bereich der Aufgabenlösung werden der Programmzweck und die Formulierung der zugehörigen Programmieraufgabe angegeben. Hierbei ist wichtig, fachlich korrekt, vollständig und präzise zu formulieren.

Die Aufgabendokumentation beinhaltet eine Übersicht mit den Angaben zum Lernziel, dem Aufgabentyp und der Aufgabenkomplexität. Weitere Informationen benennen das notwendige Vorwissen und wichtige Lernhandlungen. Dabei wird deutlich, dass die Lösung der Aufgabe Lernhandlungen zum Programmverstehen (lesen und ausführen des Python-Programms) und zum präzisen schriftlichen Formulieren des Programmzwecks enthält. Hinzu kommt das Schreiben des zugehörigen Programmierauftrages als kurze und spezifische Aufgabenformulierung. Ergänzende Angaben der Aufgabendokumentation verweisen auf Lehrbuchseiten bzw. eine Webseite, in der unterstützende Information für die Aufgabenbearbeitung einsehbar sind. Wichtige abschließende Inhalte der Aufgabendokumentation umfassen den Namen des Autors, eine Versions- oder Datumsangabe sowie die Benennung der Lizenz.

````
Themenbereich: ...Zuordnung zu einem fachlichen Themenbereich...
## Lernaufgabe: ...Titel...

...textliche Formulierung mit Angabe einer Fragestellung oder Aufträgen(Operatoren) zur Aufgabenbearbeitung...

``` python
# Programmcode mit Bezug zur Aufgabenstellung


```

---

#### Beispiellösung

``` python
# Programmcode eines Lösungsbeispiels

```

...Textanworten, Abbildungen, Skizzen oder Ergebnistabellen als Lösung der Aufgabe...

| **Lernziel**                              | **Aufgabentyp**   | **Aufgabenkomplexität** |
| ----------------------------------------- | ----------------- | ----------------------- |
| ...Benennung des Lernziels der Aufgabe... | ...               | niedrig / normal / hoch |

##### Notwendiges Vorwissen

...Auflistung von Kategorien notwendigen Vorwissens...

- ...
- ...

##### Lernhandlungen

...Auflistung von Lernhandlungen die bei der Aufgabenbearbeitung vollzogen werden...
...im Fall einer bestimmten Reihenfolge der Handlung, sollte eine numerierte Aufzählung angegeben werden...
1) ...
2) ...
3) ...


##### Unterstützende Informationen

...Web-Links, Verweise auf Bücher oder Arbeitsmaterialien...

---
Autor: ...Name und Institution...
Version: ...Monat/Jahr...
Lizenz: z.B.: CC BY-SA 4.0
````

Abb. 2: Vorlage zur Aufgabendokumentation im Markdown-Format

Task Name	Title	Learning objective	Complexity	Task type
MinMaxNumber	Minimum and maximum number	loop to repeat code sections, break a loop	2 - normal	worked out example
FunctionCalculation	Calculation of function values	loop to repeat code sections	1 - low	reverse task
ArithmeticMean_v1	Arithmetic mean calculation	loop to repeat code sections	1 - low	completion task
YourIdea	Your own loop program	loop to repeat code sections	1 - low	non-specific goal task
ArithmeticMean_v2	Arithmetic mean calculation	loop to repeat code sections and break to exit a loop	2 - normal	completion task
MovingAverage	Moving average calculation	loop to repeat code sections, break a loop	2 - normal	worked out example
MiniMarket	Mini market cash point	loop to repeat code sections, break a loop	2 - normal	reverse task
MonthlyBalance_v1	Monthly balance over the year	loop to repeat code sections	1 - low	conventional task
MonthlyBalance_v2	Monthly balance over the year	loop to repeat code sections, break a loop	2 - normal	completion task
GPSTrackLength	Length of a GPS track	loop to repeat code sections, break a loop	3 - high	conventional task

Abb. 3: Übersichtstabelle der Aufgaben des Lernbereichs „Steuerung des Programmablaufs: Wiederholungen“ aus dem Aufgabenpool zur Python-Programmierung

Learning Task: What is the purpose of the code?

Read, analyse and run the given Python program. What is it's purpose?
Write the text for a related programming task.

```
# ---- initialize variables -------
sum_price = 0.0       # total price
sum_vat = 0.0         # total VAT

# ---- loop to enter user data and cumulate values -----
while True:
    name = input("Product: ")                # enter product name as string
    if name == "":                           # check for product name
        break                                # break loop if no product name is given

    price = float(input("Price  : "))        # enter price as float
    pcs = int(input("Pieces : "))            # enter pieces as integer
    vat = float(input("VAT %  : "))          # enter VAT as float

    sum_price = sum_price + price*pcs        # cumulate prices
    sum_vat = sum_vat + price*pcs*vat/100.0  # cumulate VAT

    # print cumulated values
    print(sum_price, "EUR plus ", sum_vat, "EUR VAT \n")

print("The end :-)")
```

Solution

Purpose of the program:
The program can be used as a mini market cash point calculator.

Programming task
Write a Python program to be used as a mini market cash point calculator.
The user will enter product name, price, number of pieces and VAT for a series of products and the program will calculate the total price and the total VAT.

Learning objective	Task type	Complexity
loop to repeat code sections, break a loop	reverse task	2 - normal

Previous Knowledge

vcp-1, vcp-2: print, input, variable
branch-1: if-statement
loop-2: while-loop including break-statement

Learning Activities

1. read, run and understand the Python code
2. explain the purpose in a short and specific written statement.
3. write a short and specific programming task, that yields the given Python code

Supporting information

tutorialspoint.com: while-loops
tutorialspoint.com: break-Statement
Matthes, E. (2019). Python crash course a hands-on, project-based introduction to programming (2nd edition). No Starch Press.: Chapter 7, pages 118-121

www.python-kurs.eu: Schleifen
Theis, T. (2017). Einstieg in Python. In Rheinwerk Computing (5., aktualisierte Auflage). Rheinwerk Verlag GmbH.: Kapitel 3, Seiten 60-62

Author: Robert Ringel, Faculty Informatics/Mathematics, HTWD – University of Applied Sciences
Version: 02/2025
License: CC BY-SA 4.0

Abb. 4: Beispiel der Aufgabendokumentation aus dem Lernbereich „Steuerung des Programmablaufs: Wiederholungen“

Diese Art der Aufgabendokumentation unterstützt die effiziente Erstellung und Kuratierung eines Aufgabenpools und ermöglicht den niedrigschwelligen Zugang zum gewählten digitalen Medium. Durch die flache, gut geordnete Struktur mit wenigen, methodisch bedeutsamen Merkmalen gelingt ein schnelles Auffinden gesuchter Informationen.

5. Weiterentwicklung und Nachnutzung in anderen Fachdomänen

Die digitale Sammlung von Lernaufgaben zur Python-Programmierung eröffnet die Möglichkeit zur methodischen Gestaltung von aufgabenbasierten Programmierlehrgängen in Anlehnung an 4C/ID und Ringel (2024). Sie verdeutlicht die Vielfalt möglicher Aufgabenstellungen mit Blick auf unterschiedliche praktische Probleme unter Nutzung verschiedener Aufgabentypen. Die genannten Beispiele dienen als Anregung und Orientierung zur Gestaltung eigener Lernaufgaben, wobei aufgezeigt wurde, dass es wesentlich mehr Aufgabenarten gibt als die klassische Gegeben-Gesucht-Aufgabe. Aus der methodischen Perspektive zeigt die vorgestellte Programmieraufgabe neben dem zugehörigen Lösungsbeispiel auch die unterschiedlichen Lernhandlungen. Die Identifikation der Lernhandlungen ist das wesentliche didaktische Element der Aufgabengestaltung, da eine Lernaufgabe nur das Mittel repräsentiert, mit dessen Hilfe die Lernprozesse in Form von Lernhandlungen durchlaufen werden.

Zudem zeigt die Aufgabensammlung beispielhaft, wie Lernaufgaben dokumentiert und geordnet werden können, um sie in GitHub frei zugänglich zu machen. Die Nutzung von GitHub und Markdown-Dateien ist eine Lösung, die hohe gestalterische Freiheit (Kombination von Text, Tabellen, Grafiken, Weblinks) mit minimalem medientechnischen Gestaltungsaufwand verbindet, da die Markdown-Dateien mit Standardtexteditoren auf allen verbreiteten Computersystemen editierbar sind. Somit wird anhand des gezeigten Beispiels der Aufgabendarstellung deutlich, dass das Konzept der Aufgabendokumentation mit Markdown auch für andere Fachbereiche außerhalb der Informatik bis hin zur beruflichen Ausbildung (Körndle & Proske, 2023) oder zum Fremdsprachenlernen (Willis, 2004) nutzbar sein kann.

Um eine zügige und zielführende Orientierung in großen Aufgabensammlungen zu gewährleisten, bieten grafische Darstellungen eine gute Unterstützung. Im aktuellen Aufgaben-Pool ist dazu eine manuell erzeugte Darstellung der inhaltlichen Struktur enthalten. Diese Darstellung sollte durch automatisch erzeugte Grafiken unter Nutzung von Metainformationen (z.B. Vorkenntnisse, Lernziele, Komplexität) ergänzt werden, um eine fortwährende Aktualität zu gewährleisten. Hasse-Diagramme und Werkzeuge wie Graphviz (The Graphviz Authors, 2021) gilt es dafür zu evaluieren. Eine fachliche Einordnung dazu findet sich im Konferenzbeitrag von Körndle und Krauße (2005).

Der konkret für das Programmierenlernen angelegte Aufgaben-Pool in einem GitHub-Repository ist generisch konzipiert, so dass – im Sinne des D2C2-Projekts der Hochschuldidaktik Sachsen – eine Übertragung in andere Fachdisziplinen möglich wird. Der Aufgaben-Pool demonstriert, wie digitale Aufgabensammlungen einzelner Fachgebiete als öffentliche Bildungsressourcen frei zugänglich publiziert werden können. Für Nutzerinnen und Nutzer besteht die Möglichkeit, durch die Einreichung eigener Aufgabendokumentationen zur Erweiterung des Aufgabenpools beizutragen. Dies betrifft sowohl die Erweiterung der bestehenden Abschnitte des Aufgabenpools als auch die Neugestaltung weiterer Themenbereiche zum Programmierenlernen.

Literatur

4C/ID (2025, März). https://www.4cid.org/

Astleitner, H. (2006). *Aufgaben-Sets und Lernen. Instruktionspsychologische Grundlagen und Anwendungen.* Frankfurt am Main; Berlin; Bern; Wien [u.a.]: Lang.

Bloom, B. S. (1956). Taxonomy of educational objectives: the classification of educational goals; Handbook I: Cognitive domain. In M. D. Engelhart et al. (Eds.), *Taxonomy of educational objectives: the classification of educational goals; Handbook I: Cognitive domain.* New York: David McKay.

Chen, O., Paas, F. & Sweller, J. (2023). A Cognitive Load Theory Approach to Defining and Measuring Task Complexity Through Element Interactivity. *Educational Psychology Review, 35*(2), 63. https://doi.org/10.1007/s10648-023-09782-w

Krathwohl, D. R. (2002). A Revision of Bloom's Taxonomy: An Overview. *Theory Into Practice, 41*(4), 212–218. https://doi.org/10.1207/s15430421tip4104_2

Körndle, H. & Kraße, R. (2005). Interactive Visualisation Techniques on Queries in Structured Information Spaces. In *Marktplatz Internet: Von e-Learning bis e-Payment, 13. Leipziger Informatik-Tage (LIT 2005)*. Bonn: Gesellschaft für Informatik e. V., 264–272.

Körndle, H. & Proske, A. (2023). Arbeitsplatznahe Lernaufgaben: Ihre Modellierung, Konstruktion und Einsatz in digitalen Lehr-Lernszenarien beruflichen Lernens. In *bwp@ Profil 8: Netzwerke – Strukturen von Wissen, Akteuren und Prozessen in der beruflichen Bildung. Digitale Festschrift für Bärbel Fürstenau zum 60. Geburtstag*. https://www.bwpat.de/profil-8_fuerstenau/koerndle-proske (27.01.2025)

Liepins, L. (2024). *Markdown: Syntax.* https://markdown.de/ (27.01.2025)

Merriënboer, J. (1990). Strategies for programming instruction in high school: Program completion vs. Program generation. *Journal of Educational Computing Research,* 6, 265–285.

Merriënboer, J. & Kirschner, P. A. (2018). *Ten steps to complex learning a systematic approach to four-component instructional design.* 3. Auflage. New York, Routledge, Taylor & Francis Group.

Oellers, M. & Rörtgen, S. (2024). *Kompendium: Didaktische Metadaten.* https://doi.org/10.25656/01:29235

Ringel, R. (2024). *Entwicklung und Evaluierung eines Rahmenkonzepts zum Programmierenlernen an Hochschulen.* Dresden. https://nbn-resolving.org/urn:nbn:de:bsz:520-qucosa2-933983 (27.01.2025)

The Graphviz Authors (2021). *Graphviz.* https://graphviz.org/ (27.01.2025)

Willis, J. (2004). *A framework for task-based learning.* 1. publ., 8. impr. Longman handbooks for language teachers. Harlow: Longman.

Zitiervorschlag:

Ringel, R. (2025). Lernaufgaben digital dokumentieren, sammeln und veröffentlichen. *Perspektiven auf Lehre. Journal for Higher Education and Academic Development, 4*(1), 27-36.

DOI: 10.55310/jfhead.74