Objektorientierte Programmierung in der Sekundarstufe II des Gymnasiums

Reflexion über Herangehensweisen zur Vermittlung grundlegender Programmierparadigmen

Schriftliche Arbeit zur Zweiten Staatsprüfung für das Lehramt am Gymnasium
im Fach Informatik

Hamburg, den 8. November 2017

Pamina Maria Berg

LiV
HS 16-08-Frö

Erstgutachter	Sven Alisch
Zweitgutachterin	Christina von Bremen
Hauptseminarleitung	Dr. Sven Michael Fröhlich
Fachseminarleitung Informatik	Sven Alisch
Fachseminarleitung Mathematik	Hayo Zimmermann
Datum der mündlichen Prüfung	21.12.2017

Inhaltsverzeichnis

1 Einleitung **1**

2 Ausgangssituation **2**
2.1 Systemische Rahmenbedigungen 2
2.1.1 Rahmenplan 2
2.1.2 Curriculum des Gymnasium Ohmoor 2
2.2 Inhaltliche Ziele 3
2.3 Lerngruppen 4

3 Die Praxissituation **5**
3.1 BlueJ 5
3.2 Bausteine der Unterrichtsplanung und Didaktische Entscheidungen 5
3.3 Vorgehensweise und Methodische Entscheidungen 7

4 Reflexion der Herangehensweisen **8**
4.1 Kritische Betrachtung 8
4.2 Analyse der Unterrichtssituationen 10

5 Schlussfolgerungen für die Unterrichtseinheit **14**

6 Fazit und Ausblick **17**

1 | Einleitung

Die Lehre des Programmierens ist stark von einem schrittweisen Abarbeiten von Programmierparadigmen in einer ausgewählten Programmiersprache geprägt. In Lehrbüchern werden beispielsweise klassischer Begriffe der Objektorientierung *Klasse, Objekt, Methode, Parameter* anhand von Syntaxbeispielen und ggf. kleiner Projekte vermittelt (bspw. in [Abt15] [Bar03] [Ehm09] [Ull12]). Dieses Vorgehen spiegelt sich auch in der Universitätslehre und auf E-Learning-Plattformen wider.

Informatische (Grund-)Bildung sollte als Teil der Allgemeinbildung (vgl. [Bre94]) auch allgemeine Konzepte der Informatik vermitteln. HUBWIESER weist in seinem Standardwerk zur Informatik-Didaktik auf einen jedoch immer noch aktuellen Sachverhalt hin:

> Beim Betrachten entsprechender Rahmenpläne entsteht der Eindruck, dass [...] Programmierkurse im Kleinen in diesem Unterricht durchgeführt werden ([Hub07, S.40] aus [Koe93]).

Um wirklich tragfähige Vorstellungen von Informatik bei den Schülerinnen und Schülern zu etablieren, bedarf es allerdings mehr als einem Programmierkurs, bei dem ein Großteil der Lernenden daran scheitert, das zu Lernende anzuwenden, weil sie damit beschäftigt sind, eine beliebige Syntax auswendig zu lernen (vgl. [Hum02], [Mod11]).

Die im Rahmen der Unterrichtsheit *Objektorientierte Modellierung und Programmierung* durchgeführte und im Folgenden vorgestellte Unterrichtssituation diente dem Versuch, grundlegende Programmierparadigmen als Teil informatischer Grundbildung sowohl mit Hilfe dieses klassischen Ansatzes als auch abseits vom Quellcode zu vermitteln.

Beginnend mit der Ausgangssituation werden in der vorliegenden Arbeit die Rahmenbedingungen sowie die Durchführung des Unterrichts beschrieben. Diese wird anschließend anhand von allgemein- und fachdidaktischen Kriterien analysiert und es werden Schlussfolgerungen für die Gestaltung zukünftiger Unterrichtseinheiten formuliert. Abschließend erfolgt ein Fazit sowie ein kurzer Ausblick.

2 | Ausgangssituation

Es werden nun zunächst die systemischen Rahmenbedingungen, sowie die Voraussetzungen dargestellt, die sich aus den inhaltlichen Lernzielen und den Lerngruppen ergeben.

2.1 Systemische Rahmenbedigungen

Der Unterrichtsplanung zugrunde liegen der Rahmenplan Informatik für die Gymnasiale Oberstufe (vgl. [HH09]) sowie das schulinterne Curriculum des Gymnasium Ohmoor. Im Folgenden werden die für die durchgeführte Unterrichtspraxis relevanten Inhalte kurz erläutert.

2.1.1 Rahmenplan

Der Rahmenplan Informatik für die Gymnasiale Oberstufe in Hamburg spezifiziert die *Objektorientierte Modellierung* als verbindlichen Inhalt, wobei eine explizite Forderung nach der „Erarbeitung der Sprachelemente der verwendeten objektorientierten Programmiersprache“ ([HH09, S.17]) besteht. Des Weiteren sind verschiedene Anforderungsbereiche definiert, durch die sowohl fachliche als auch überfachliche Kompetenzen erworben und überprüft werden sollen. Die für die zu untersuchende Unterrichtssituation relevante Kompetenz bezieht sich auf den Bereich des *Darstellen und Interpretieren*, in dem die Schülerinnen und Schüler „Modelle und Algorithmen sowohl grafisch als auch verbal“ beschreiben können sollen (vgl. [HH09, S.16]). Ein großer Fokus wurde planungsbedingt auch auf die Kompetenz des *Kommunizierens und Kooperierens* gelegt (siehe hierzu Abschnitt **3.3**).

2.1.2 Curriculum des Gymnasium Ohmoor

Die Fachschaft Informatik des Gymnasium Ohmoor konkretisiert im schulinternen Curriculum die allgemein formulierten Vorgaben aus dem Rahmenplan Informatik. So ist im zweiten Semester der Gymnasialen Oberstufe das Thema *Objektorientierte Modellierung/Programmierung von Grafiksystemen mit Java* angesiedelt (vgl. [Ohm16, S.6f.]). Als verbindlicher Inhalt ist

hier unter anderem die „Erarbeitung von Sprachelementen: [...] Kontrollstrukturen“ ([Ohm16, S.7]) genannt, zu denen auch das Programmierparadigma der Schleifenkonstrukte gehört.

2.2 Inhaltliche Ziele

Die inhaltlichen Ziele der Unterrichteinheit waren sowohl fachlicher als auch überfachlicher Art und lassen sich wie folgt zusammenfassen:

- Die Schülerinnen und Schüler[1] analysieren das BlueJ-Projekt, um sich mit den wesentlichen Merkmalen von Schleifen als Kontrollstrukturen in der Programmierung vertraut zu machen.
- Die SuS erläutern anhand eines Minimalbeispiels in Java-Syntax den Ablauf einer *for-*, *while-* oder *do-while*-Schleife, indem sie eine Kurz-Vorführung im Plenum vorbereiten und durchführen.

Die hierzu passenden Einzellernziele sind:

- Die SuS geben mindestens die prägnanten Merkmale des Quellcodes an und beschreiben den grundsätzlichen Programmablauf im BlueJ-Projekt und sind bestenfalls in der Lage, eigene Schleifenkonstrukte zu implementieren und die Geeignetheit des gewählten Konstrukts zu begründen.
- Die SuS beschreiben mindestens umgangssprachlich den Zusammenhang zwischen dem von ihnen gewählten Schleifenkonstrukt und ihrer Präsentation und beurteilen bestenfalls die Passung von Präsentation und Schleifenkonstrukt der eigenen und anderen Gruppen.
- Die SuS stellen mindestens auf Nachfrage die wesentlichen Unterschiede der drei Schleifenkonstrukte dar und vergleichen diese bestenfalls im Hinblick auf verschiedene Einsatzszenarien.

[1] Im Folgenden SuS genannt

2.3 Lerngruppen

Der in dieser Unterrichtseinheit untersuchte Lerngegenstand wurde in zwei Vergleichsgruppen mit methodisch variierten Vorgehensweisen vermittelt. Diese werden nachfolgend als Vergleichsgruppe **A** und **B** bezeichnet.

Vergleichsgruppe **A**:
Der Informatik-Wahlpflichtkurs auf grundlegendem Niveau umfasst insgesamt 17 Schülerinnen und Schüler. Die Leistungsspanne ist relativ groß, da es mehrere SuS gibt, die sehr progammieraffin sind und sich auch außerhalb des Unterrichts mit Programmierung beschäftigen. Einer der SuS setzt sich leistungstechnisch deutlich von der Gruppe ab, da er durch sein Praktikum bereits umfassende Programmierkenntnisse in Java besitzt und diese selbstständig und mühelos im Unterricht umsetzen kann. Im Gegensatz dazu, gibt es auch SuS, die versuchen, sich bei Arbeitsphasen aus dem Unterrichtsgeschehen herauszuziehen und dadurch größere Schwierigkeiten besitzen, Quellcode zu verstehen, zu bearbeiten oder zu produzieren. Insgesamt fällt schwächeren SuS das Programmieren grundsätzlich noch etwas schwerer.

Vergleichsgruppe **B**:
Die zweite Vergleichsgruppe ist der PGW-Profil-Kurs, bestehend aus 26 Schülerinnen und Schülern, deren Leistungsniveau sich eher heterogen gestaltet, wobei sich im Vergleich zu Gruppe **A** keine echte Leistungsspitze abzeichnet. Das Vorwissen der SuS in Bezug auf die zu untersuchende Unterrichtseinheit war (bedingt durch verschiedene Projektphasen der SuS in anderen Fächern) etwas geringer als bei der Vergleichsgruppe **A**, jedoch nicht in diesem Maße, als dass eine Lernhürde bei der Analyse von Quelltext zu erwarten gewesen wäre. Insgesamt ist anzumerken, dass der Großteil der Gruppe zwar motiviert und konzentriert an Programmieraufgaben arbeitet, im Vergleich zur Gruppe **A** jedoch anteilig gesehen weniger SuS sicher mit der Java-Syntax umgehen können.

Beide Vergleichsgruppen waren bereits sicher im Umgang mit der Entwicklungsumgebung *BlueJ* und hatten in unterschiedlichen Kontexten eigene Programmiererfahrungen mit Java machen können. Größere Schwierigkeiten beim eigenständigen Umgang mit Quellcode waren deshalb nicht zu erwarten.

3 | Die Praxissituation

Im Folgenden werden die für die Planung der Praxissituation maßgeblichen Details und Entscheidungen, sowie die Vorgehensweisen für die Durchführung beschrieben.

3.1 BlueJ

Die Java-Entwicklungsumgebung BlueJ wurde an der Monash University in Australien mit dem Ziel entwickelt, eine Umgebung für Programmieranfänger mit einer einfachen Benutzerschnittstelle zu schaffen (vgl. [Bar03, S.14]). BARNES und KÖLLING betonen die besondere Geeignetheit von BlueJ in der Lehre und führen diese auf die native Visualisierung der Klassenstruktur und die damit einhergehende Möglichkeit, mit den Objekten direkt zu interagieren, „ohne Testklassen schreiben zu müssen“ (vgl. [Bar03, S.15]). Es handelt sich hierbei um eine vollwertige Entwicklungsumgebung, die auf dem aktuellen Java Development Kit (JDK) läuft und als Compiler und virtuelle Maschine (JVM) Software der Firma Oracle (bis 2010 Sun Microsystems) verwendet (ebd.).

3.2 Bausteine der Unterrichtsplanung und Didaktische Entscheidungen

Unter „Programmieren“ im klassischen Sprachgebrauch wird immer ein Am-PC-Sitzen und Quellcode schreiben verstanden. Die Informatik und insbesondere der Teilbereich der Objektorientierten Programmierung hat jedoch in weiten Teilen der Lehre bereits den Anspruch, syntaxübergreifende Konzepte anstatt primitivem (Programmier-)Sprachenverständnis zu vermitteln (vgl. hierzu den *Objects-First*-Ansatz von BARNES und KÖLLING [Bar03]). Aus diesem Grunde wurden die SuS bei der Unterrichtseinheit angeregt, den Abstraktionsschritt vom Quelltext zum spielerischen physischen Ausprobieren des Programmierparadigmas *Schleifen* zu vollziehen. Sie sollten das Schema eines Schleifenkonstrukts nicht nur verstehen, sondern auch nachempfinden und – mit einem Ausblick auf das folgende Semesterthema *Algorithmen und Datenstrukturen* – ein Gefühl dafür entwickeln, dass mehrere Einzelschritte

für die Ausführung der Anweisung bis zum Ergebnis notwendig sind. Die Unterrichtseinheit zum Lerngegenstand Schleifenkonstrukte besteht aus zwei Bausteinen, die inhaltlich ineinandergreifen, jedoch didaktisch und methodisch andere lerntheoretische Ansätze verfolgen.

Der erste Baustein besteht aus einem BlueJ-Projekt, welches ich durch meine Tätigkeit als Übungsgruppenbetreuerin an der Universität Hamburg kennengelernt habe. Dieses Projekt ist an das Lehrbuch zur Objektorientierten Programmierung mit Java und BlueJ von BARNES und KÖLLING [Bar03] angelehnt und versucht, über kurze Methodenrümpfe, die von den Lernenden analysiert werden sollen, verschiedene Arten von Schleifenkonstrukten zu vermitteln.
Zusätzlich zu diesem Kurzprojekt haben die SuS ein Arbeitsblatt mit verschiedenen Aufträgen bekommen, welches sie schrittweise durch das Projekt führen sollte (vgl. Anhang **A**).

Anhand der Anlage des Projekts ist zu erkennen, dass dieser Aufgabenstellung ein eher *kognitivistisches* lerntheoretisches Fundament zugrunde liegt. Die SuS versuchen, in einer Art „Dialog“ mit dem Quellcode zunächst den Lerngegenstand zu verstehen (Aufgabenteile 1–3), und darauf aufbauend eine Aufgabe, bzw. ein Problem zu lösen (Aufgabenteil 4). Der Lerngegenstand wurde vorstrukturiert und auch die Problemlösestrategie wurde innerhalb des BlueJ-Projekts bereits vorgemacht (vgl. [Schwa07, S.219]).

Den zweiten Baustein haben sich die SuS jeweils eigenständig erarbeitet: Auf Grundlage ihrer basalen Syntaxkenntnisse haben die SuS in Gruppen verschiedene leere Schleifenkonstrukte in Java bekommen, deren Ablauf sie jeweils zunächst analysieren und dann in Form einer interaktiven Gruppenpräsentation darstellen sollten. Hierbei war es den SuS freigestellt, welche Hilfsmittel sie dazu einsetzen und welche Aktivitäten vorgeführt werden können. Es sollte lediglich deutlich werden, wie die von ihnen vorgestellte Sequenz von Anweisungen mit der von ihnen ausgewählten *for-, while-* oder *do-while*-Schleife im Zusammenhang steht.

Der zweite Baustein stellt Wissen als Rohmaterial zur Verfügung und baut auf die Interaktion der Lernenden miteinander. Die Lehrkraft bekommt bei der Vorführung die Rolle eines Moderators und lässt die SuS ihr Wissen selbstaktiv konstruieren, sowie über ihr Handeln reflektieren. Daher entspricht dieser Ansatz eher der Lerntheorie des *Konstruktivimus* (vgl. [Schwa07, S.219]).

3.3 Vorgehensweise und Methodische Entscheidungen

Die beiden Bausteine wurden zum Zweck eines produktiven Vergleichs in jeweils umgekehrter Reihenfolge mit den Vergleichsgruppen durchgeführt. So hat Vergleichsgruppe **B** zunächst mit der interaktiven Vorführung, also dem enaktiven Ansatz, begonnen und sich danach mit dem BlueJ-Projekt beschäftigt – der Vergleichsgruppe **A** wurde zuerst der Quelltext und das Arbeitsblatt ausgegeben, mit deren Erarbeitung sie in Einzel- oder Partnerarbeit und ohne weitere Einhilfen begonnen haben.

Es gab jeweils nach der Erarbeitung der Bausteine, die jeweils eine Doppelstunde in Anspruch genommen hat, eine Besprechungsphase, in der die SuS über ihr Vorgehen reflektiert und sich im Plenum ausgetauscht haben. In dieser Phase wurden außerdem die wichtigsten Punkte des erarbeiteten Lerngegenstands in Bezug auf die Syntax oder die Vorführung nochmals gemeinsam herausgearbeitet.

Zweck der Variation der Vorgehensweise war mein Interesse an der Effektivität und dem Einfluss des Arbeitsauftrags zur freien Präsentation der Schleifenkonstrukte, also des Lernens durch *Bewegungshandlungen* (vgl. [Aeb11, S.183f.]) auf das Grundverständnis der SuS bezüglich des Programmierparadigmas.

4 | Reflexion der Herangehensweisen

Bei der Reflexion der Herangehensweisen und der Praxissituationen wird es nun darum gehen, die beiden in ihrer Reihenfolge variierten Vorgehensweisen kritisch zu betrachten, in Bezug zueinander zu setzen und eine daraus resultierende Fragestellung zu entwickeln und zu untersuchen.

4.1 Kritische Betrachtung

Erarbeitung Gruppe **A**: Die erste Vergleichsgruppe hat mit dem BlueJ-Projekt begonnen. Sie sollten hierbei ohne Einhilfen arbeiten und es wurden Fragen nur in einem sehr geringen Maße beantwortet.

Die analysierenden Aufgabenteile (1–3) wurden von den SuS ohne Probleme bearbeitet, bei Aufgabe 4 trat jedoch eine Lernhürde auf: Die SuS wurden dazu aufgefordert, den Lernschritt vom *Verstehen* des Lerngegenstands zum *Anwenden* des Konzepts auf eine neue Situation zu tätigen. Viele der SuS haben sich an die Aufgabe herangewagt und Schritt für Schritt versucht, eine Lösung zu implementieren, es waren aber auch einige SuS wahrzunehmen, für die das Implementieren eigener Ideen immer noch eine Schwierigkeit darstellte. So gab es während der Arbeitsphase sehr gegensätzliche Schülerkommentare:

- „Ich weiß gar nicht, wie ich jetzt anfangen soll."
- „Meine *Switch*-Anweisung funktioniert noch nicht richtig, aber der Rest klappt schon ganz gut."

Eine deutlich größere Motivation auf Seiten der SuS war in der Unterrichtsstunde mit der Erarbeitung einer Vorführung zu den Schleifenkonstrukten festzustellen. Die SuS haben mit sehr viel Engagement verschiedenste Präsentationen erarbeitet und gezeigt, dass sie die Konzepte der drei Schleifentypen grundsätzlich gut verstanden haben. Auch war bei dieser Methodenwahl zu sehen, dass sowohl die schwächeren als auch die stärkeren SuS ihre Ideen gleichermaßen gut einbringen und umsetzen konnten. Der zweite Lernansatz, der in dieser

Vergleichsgruppe eher einen vertiefenden und überprüfenden Charakter hatte, fand bei SuS verschiedener Leistungsbereiche Anklang.

Auf Nachfrage, ob die Vorführungen als lernförderlich oder lernhinderlich wahrgenommen wurde, bekam ich von der gesamten Gruppe ein positives Feedback und insbesondere von SuS, die bei den BlueJ-Aufgaben an ihre Grenzen gestoßen sind, wurde angemerkt, dass die vorgestellten konkreten Beispiele ihr Verständnis der Schleifenkonstrukte gefördert haben.

Erarbeitung Gruppe **B**: Die zweite Vergleichsgruppe begann die Unterrichtseinheit mit der Entwicklung und Vorführung der Schleifenkonstrukte. Auch in dieser Gruppe war die Motivation zur Bearbeitung der Aufgabe sehr groß, so dass auch in dieser Lerngruppe alle Mitglieder der Kleingruppen eine aktive Rolle in der Präsentation ihrer Ergebnisse eingenommen haben.

Die SuS dieser Gruppe waren der Aufgabenstellung gegenüber merklich offener und schienen einen besseren Zugang zum Lerngegenstand zu haben als bei den vorangegangenen Unterrichtsstunden, in denen eher mit Projekten vom Typ des ersten Bausteins gearbeitet wurde. Die Entfernung von der Syntax und dem Computer selbst war für die SuS eine willkommene Abwechslung und hat die Herausforderung, mit einem neuen Programmierparadigma in Berührung zu kommen, wesentlich erleichtert.

Die Bearbeitung des BlueJ-Projekts stellte jedoch in dieser Gruppe für viele SuS wieder eine Herausforderung dar. Die SuS kamen mit dem Analysieren des Quellcodes zurecht, jedoch wurde von kaum einer/m der SuS die vierte Aufgabe erfolgreich bearbeitet.

Zur Reihenfolge der Bausteine gab es sehr unterschiedliche Schüleraussagen. Einige empfanden den Einstieg in den Lerngegenstand durch das handelnde Lernen als sehr anregend und förderlich, um hinterher in den Quellcode einzusteigen – für manche SuS stellte die Programmieraufgabe kein größeres Problem dar und trotzdem sind Rückmeldungen wie die folgenden in der Nachbesprechung geäußert worden:

- „Das hat mir geholfen, nochmal zu sehen, ob ich das alles richtig verstanden habe.“
- „In BlueJ war das ja nicht so deutlich zu sehen.“

4.2 Analyse der Unterrichtssituationen

Allgemeindidaktische Aspekte

(Guter) Unterricht und dessen Durchführung ist ein Aspekt des Lehrerhandelns, der sowohl in neueren als auch älteren Publikationen zu allgemein- und fachdidaktischen Themen zu finden ist. MEYER hat sich ausführlich mit dem Thema *Was ist guter Unterricht?* beschäftigt (vgl. [Mey04]), und die entwickelten Gedanken wurden unter anderem von BARZEL ET AL. als Kriterien für guten Unterricht unter verschiedenen Gesichtspunkten festgehalten (vgl. [Bzl16, S.24f.]):

1. ***Methoden*: Methodenvielfalt und -variabilität**
2. ***Fachliche Prozesse*: Inhaltliche Klarheit**
3. ***Heterogenität*: Individuelles Fördern**
4. *Bewertung*: Transparente Leistungserwartung
5. *Kommunikation*: Gesprächskultur
6. ***Verantwortung/Kooperation*: Verantwortungsübernahme**
7. ***Vernetzung/Sinnstiftung*: vertikale Vernetzung, passgenaues, gezieltes Üben**

In Bezug auf die *Methoden* ist festzustellen, dass in Bezug auf die Vermittlung des Lerngegenstands *Schleifenkonstrukte* eine Variation der Vorgehensweisen vorhanden war. Anzumerken ist aber auch, dass die Methode der Bearbeitung des BlueJ-Projekts jedoch ein für die SuS gewohnter und in den vorangegangenen Stunden immer wieder durchgeführter Arbeitsauftrag war. Die Anlage der beiden Methoden bietet generell eine geeignete Variablität für den Einsatz in dieser Unterrichtseinheit, da auch andere Programmierparadigmen derart vermittelt wurden[2] oder in Zukunft vermittelt werden können (vgl. Abschnitt **5**).

Die *inhaltliche Klarheit* war in dem BlueJ-Projekt deutlich erkennbar, da auch die Arbeitsaufträge auf den Quellcode zugeschnitten waren. Bei der enaktiven Erarbeitung war ein hoher Grad an Offenheit in mehreren Dimensionen erkennbar, so dass die SuS sowohl auf dem *Weg* zu ihrem Ergebnis als auch bei dem *Ziel* allein handeln und entscheiden mussten. Dies hat dazu geführt, dass es zu Beispielen von Schleifenkonstrukten kam, die zwar von den SuS klar zu einem Typ zugeordnet werden konnten, aber auch Raum für Diskussion zu spezifischen

[2]Die BlueJ-Projekte wurden bei der Durchführung dieser Unterrichtseinheit standardmäßig eingesetzt

Randfällen geöffnet haben. Die Präsentationen stellten eher Spezialfälle von Schleifen dar, so dass an dieser Stelle eine Nachjustierung des Arbeitsauftrags stattfinden müsste. Der Grad der Offenheit hat allerdings auch einen guten Spielraum für die *individuelle Förderung* der SuS eröffnet: Schwächere SuS konnten einen informatischen Sachverhalt für sich greifbarer machen und erleben, stärkere SuS haben dabei überfachliche Sozialkompetenzen trainiert und auf fachlicher Ebene diskutiert. Dies hat gleichzeitig auch die *Verantwortungsübernahme* einiger SuS gestärkt. Anzumerken ist jedoch, dass für eine sinnvolle Förderung dieses Aspekts eine gezielte, eher homogen strukturierte Einteilung der Arbeitsgruppen notwendig wäre, um auch bei leistungsschwächeren SuS ein Verantwortungsgefühl für ihr Arbeitsergebnis hervorzurufen.

Bei der *vertikalen Vernetzung* und dem *gezielten Üben* ist festzustellen, dass einerseits durch den Einsatz des BlueJ-Projekts ein Anknüpfen an bereits vorhandenes Wissen zur Programmierung mit Java stattfinden konnte, und andererseits sowohl die Programmierung als auch die Vorbereitung der Präsentation zum Thema ein – in Abhängigkeit der Reihenfolge in der Durchführung des Unterrichts in unterschiedlich starker Ausprägung – gezieltes Üben für die SuS ermöglicht hat.

Ergänzend sind noch einige der acht Prinzipien didaktischen Handelns nach HUBWIESER erwähnenswert, da diese einen weiteren kritischen Blickwinkel eröffnen (vgl. [Hub07, S.15ff.]):

Die *Motivation* der SuS war bei den beiden Unterrichtsbausteinen unterschiedlich stark ausgeprägt. Während die Erarbeitung einer Präsentation insgesamt zu einer hohen Schülermotivation führte, war bei der Arbeit mit dem BlueJ-Projekt eine eher verhaltene Stimmung wahrzunehmen. Diejenigen, die schon zu früheren Zeitpunkten gern mit dem Quellcode gearbeitet haben, empfanden die Aufgabe als eine kleine Herausforderung, um zu überprüfen, wie weit sie nun gekommen waren. Es war bei den nicht-programmieraffinen SuS aber auch eine unbefriedigend schwache Motivation festzustellen. Dies könnte zum einen daran gelegen haben, dass diese SuS generell Schwierigkeiten im Umgang mit Quellcode hatten, aber auch daran, dass sie sich insbesondere mit Aufgabe 4 überfordert fühlten und so in eine resignierte Arbeitshaltung übergegangen sind.

Die Präsentation der Schleifenkonstrukte hat zur *Kreativitätsförderung* der SuS beigetragen, da ein abstrakter informatischer Programmieraspekt sinnvoll veranschaulicht werden sollte. Die o.g. Offenheit des Ergebnisses hat dazu geführt, dass die SuS kreative Lösungsideen anbringen, diskutieren und umsetzen konnten. Diesem Anspruch konnte das BlueJ-Projekt

leider nicht genügen und es stellt sich bereits an diesem Punkt die Frage, in wie weit die Aufgabenstellung sinnvoll verändert werden könnte[3].

Fachdidaktische Aspekte

HUBWIESER entwickelt in seinem Standardwerk zur Didaktik der Informatik Forderungen an den Informatikunterricht, die insbesondere methodischer Art sind und an denen sich nun die kritische Auseinandersetzung der Unterrichtssituation fachdidaktisch orientieren wird ([Hub07, S.67]:

1. „Erzeugung einer entspannten Arbeitsatmosphäre“
2. **„Einordnung der Lerninhalte in größere Sinnzusammenhänge [...], um den Schülern die Bildung präpositionaler Netzwerke zu ermöglichen“**
3. **„Förderung einer aktiven Auseinandersetzung mit dem Stoff, wobei unbedingt vor der Präsentation von Lösungen ein ausreichendes Problembewusstsein erzeugt werden muss.“**
4. **„Anbieten verschiedener Perspektiven und Zugänge zum selbst Thema im Sinne der *Cognitive Flexibility*“**
5. **„Erzeugung möglichst authentischer Problemsituationen, um mit den Schülern Problemlöseverhalten in einer Umgebung zu trainieren, in der dieses Verhalten tatsächlich auch benötigt wird.“**
6. **„Altersgemäße Darbietung der Lerninhalte“**

Eine zentrale Schwierigkeit der Unterrichtseinheit *Objektorientierte Programmierung und Modellierung* ist die Wahl eines geeigneten Kontexts. Es existieren verschiedene Überlegungen und Unterrichtskonzepte, wie unter anderem der *Raumplaner* von ALBOWSKI [Alb]. Die beschriebene Unterrichtssituation hat den Lerngegenstand vom Kontext losgelöst vermittelt, so dass kein größerer Sinnzusammenhang innerhalb des gewählten Semesterkontextes hergestellt werden konnte. Da Sinnzusammenhänge auch auf innerfachlicher Ebene gebildet werden können, kann hierzu jedoch angemerkt werden, dass die SuS eine Sinnhaftigkeit des

[3] Eine sinnvolle Veränderung der Aufgabenstellung bezieht auch den Aspekt mit ein, dass insbesondere die schwächeren SuS sich nicht „allein gelassen“ fühlen und trotzdem ein gewisses Maß an Offenheit vorhanden sein muss.

Lerngegenstands auf der reinen Programmierebene herstellen konnten. Für die SuS waren die Vorteile einer Vermeidung von Coderedundanzen leicht zu erkennen.

Die SuS wurden angeregt, sich mit Hilfe *unterschiedlicher Zugänge* mit dem Lerngegenstand auseinanderzusetzen. Es wurde eine hohe Aktivität der SuS gefordert und sie haben das Thema sowohl auf symbolischer als auch enaktiver Ebene betrachtet. Im Sinne der Punkte drei und vier wurde demnach eine *aktive Auseinandersetzung* mit dem Stoff gefördert. Kritisch zu sehen ist der Aspekt des Herstellen eines *ausreichenden Problembewusstseins* (vgl. Punkt 3). Die SuS haben in vorangegangenen Unterrichtsstunden mehrfach mit BlueJ-Projekten gearbeitet – es wurde jedoch nicht explizit auf Coderedundanzen o.ä. eingegangen. Damit einhergehend kann Punkt fünf aufgegriffen werden: Die Aufgaben der SuS waren zum einen produktorientiert, zum anderen offen gestellt. Der Fokus lag auf losgelöstem konzeptuellem Wissen, so dass weder eine *authentische Problemsituation* geschaffen noch die *Problemlösefähigkeit* der SuS trainiert wurde.

Die Lerninhalte wurden zum einen mit einer fachlich klassischen Darstellung (mit Hilfe von Quellcode in einer Entwicklungsebene) zur Verfügung gestellt, zum anderen wurde eine Darbietung durch die SuS mit einer interaktiven Präsentation eingefordert. Erstere kann als durchaus *altergemäß* angesehen werden, da insbesondere im Rahmenplan der BSB explizit die „Erarbeitung der Sprachelemente“ sowie die „Nutzung einer IDE mit UML-Diagrammen und Quellcode zur schrittweisen Implementierung eines Informatiksystems“ [HH09, S.17] gefordert wird. Zur Darbietung der SuS ist anzumerken, dass diese den Versuch darstellte, Programmierparadigmen *unplugged* – also vom Computer unabhängig – zu vermitteln. Es können sowohl für- als auch widersprechende Argumente zur *Angemessenheit der Darbietungen* gefunden werden. Auf eine weiterführende Betrachtung möglicher Verbesserungen wird in Abschnitt **5** eingegangen.

5 | Schlussfolgerungen für die Unterrichtseinheit

Auf Grundlage der Untersuchung und Reflexion der Praxissituation werden nun die daraus resultierenden Konsequenzen und Fragestellungen, sowie die hieraus ableitbaren Forderungen an eine verbesserte Unterrichtskonzeption dargestellt.

Betrachtet man die Unterrichtssituation zur Vermittlung von Schleifenkonstrukten, so lassen sich in direktem Bezug hierzu einige Konsequenzen formulieren: Die Präsentationsform an sich war methodisch gesehen erfolgreich und hat den SuS einen weiteren Zugangspunkt zum Lerngegenstand eröffnet, so dass diese Arbeitsweise als Basisidee für die Weiterführung der Unterrichtskonzeption geeignet ist.

Die Aufgabenstellung zur Präsentation sollte zunächst ein konkretes Beispiel liefern, um, anstelle von Spezialfällen, verallgemeinerbare Situationen anzuregen. Zur Beibehaltung des Vorteils der individuellen Förderung kann die Aufgabenstellung erweitert und eine frei ausgedachte Situation zum Durcharbeiten des Lerngegenstands (vgl. [Aeb11]) als zweiter Schritt gefordert werden.

Um den Ansprüchen eines fachdidaktisch fundierten Unterrichts zu genügen, sollte bei einer verbesserten Unterrichtskonzeption zum gewählten Lerngegenstand eine Motivation aus einer authentischen Problemsituation heraus gefunden und an die SuS weitergetragen werden (s. [Hub07, S.68]).

Die Auswahl des BlueJ-Projekts zur Bearbeitung für die SuS hat Lernhürden auf verschiedenen Ebenen erzeugt (vgl. **4.1**). Diese müssten entweder durch Anpassung der Aufgabenstellung oder eine stärkere Lernbegleitung und -unterstützung auf Seiten der Lehrkraft überwindbarer gestaltet werden.

Aus der durchgeführten Praxissituation und der ausführlichen Analyse hat sich für mich die folgende Fragestellung ergeben:

Fällt den Schülerinnen und Schülern der Umgang mit Programmiersprachen und -konzepten leichter, wenn neben der (theoretischen) Vermittlung auch eine Handlungsorientierung abseits des Computers stattfindet?

Mit Rückbezug auf die Erkenntnisse aus **4.2** können nun weitere Überlegungen zu einer möglichen Neukonzeption der Unterrichtseinheit zur *Objektorientierten Programmierung und Modellierung* unter Berücksichtigung o.g. Fragestellung formuliert werden:

Die durch die Erarbeitungsform einer interaktiven Präsentation merklich erhöhte Motivation der SuS, sich mit dem gewählten Lerngegenstand auseinanderzusetzen, sollte als Denkanlass gesehen werden, um auch andere Zugangsweisen beim Unterrichten von Objektorientierter Programmierung und Modellierung auszuprobieren.

Anstatt SuS sich nur durch das Betrachten und Analysieren von Quellcode grundlegende Programmierparadigmen erarbeiten zu lassen, kann eine enge Verknüpfung zwischen dem Programmieren am Computer und dem echten Handeln im Sinne der verwendeten Methode eine Chance bieten, um den SuS sowohl mehrgliedrige Zugänge zu den Lerngegenständen zu ermöglichen, als auch Lernhürden in Bezug auf das Erlernen einer Programmiersprache besonders bei SuS ohne Vorkenntnisse abzubauen.

Um grundlegende Programmierparadigmen bei den SuS zu tragfähigen Grundkonzepten auszubilden, sollte der vertikalen Vernetzung des Wissens ein besonderer Fokus zugetragen werden. Dies beginnt bei der Wahl eines geeigneten Kontextes, der sowohl die inhaltlichen Bedingungen des Rahmenplans erfüllt, als auch an verschiedenen Stellen echte Problemsituationen ermöglicht (vgl. Punkte 3 und 5 nach [Hub07, S.67]).

In der Konkretisierung könnte in Anlehnung an den existierenden Kontext *Raumplaner* beispielsweise folgendermaßen nachjustiert werden:

Innerhalb des Kontexts *Raumplaner* wäre eine übergeordnete Aufgabenstellung für das gesamte Halbjahr denkbar, die verschiedene Problemstellungen hervorruft. Dies käme einem Großprojekt über mehrere Wochen gleich, bei dem die SuS von sich aus auf Problemsituationen stoßen, für deren Lösung sie sich unterschiedliche Programmierfähigkeiten erarbeiten müssen[4]. Auf diese Weise würden die SuS innerhalb authentischer Problemsituationen ihre Problemlösefähigkeiten in einer realitätsnahen Umgebung trainieren (vgl. Punkt 5 nach [Hub07, S.67]).

[4]Hieraus würde sich eine modular aufgebaute Werkstattarbeit ergeben, die allerdings gerade zu Beginn der Lerneinheit zu Schwierigkeiten führen könnte.

Grundlegende Gestaltungsideen könnten nicht nur direkt in BlueJ (oder einer anderen Entwicklungsumgebung), sondern beispielsweise durch Mock-Ups auf Papier ausprobiert werden. Dies bietet unter anderem den Vorteil, dass die SuS *Objekte* als etwas Greifbares erfahren, und sie diese beispielsweise durch Ausschneiden zunächst mühsam *erzeugen* müssen. Die Grundidee der *Klassen* als *Baupläne* der Objekte würde in diesem Zusammenhang von den SuS konkret erlebt werden können (s. hierzu [Aeb11, S.109]).

Die Handlungsorientierung kann nicht nur für Schleifenkonstrukte als Idee genutzt werden: Auch weitere Kontrollstrukturen wie *Sequenzen* und *Fallunterscheidungen* können von den SuS sichtbar gemacht werden, indem ein Teil einer Programmierung aktiv durchgespielt wird.

Durch die allgemein gehaltenen Schlussfolgerungen an die Vermittlung von Objektorientierter Programmierung und Modellierung in der Oberstufe, kann eine derartige Konkretisierung auch auf andere Kontexte, wie zum Beispiel die Spieleentwicklung mit BlueJ oder Greenfoot, übertragen werden.

6 | Fazit und Ausblick

Die vorliegende Arbeit beschäftigt sich mit der Frage, ob *Bewegungshandlungen* im Sinne von AEBLI zu einem besseren Zugang und Verständnis von grundlegenden Programmierparadigmen führen können. Hierzu wurde eine konkrete Durchführung dieser Grundidee aufgegriffen, beschrieben und unter allgemein- und fachdidaktischen Aspekten analysiert. Die Auseinandersetzung führte zu einem konkreten Überdenken bestehender Unterrichtskonzepte und dem Formulieren weiterführender Forderungen an die Vermittlung von Objektorientierter Programmierung im Informatikunterricht in der Sekundarstufe II des Gymnasiums.

Der Wechsel von Darstellungsformen grundlegender Programmierparadigmen im weiteren Sinne bietet die Chance, die Heterogenität von Lerngruppen in der Oberstufe produktiv zu nutzen. Es wurde deutlich, dass die Handlungsorientierung SuS aller Leistungsgruppen gleichermaßen motiviert. Es werden sowohl Lernanlässe für leistungsschwächere SuS geschaffen, als auch ein Durchdenken von Randfällen und das Beurteilen der Geeignetheit entwickelter Beispiele für leistungsstarke SuS ermöglicht. Dies kann z.B. in einer anschließenden Besprechung und Reflexion der Präsentationsphase stattfinden.

Auf die in **5** formulierte Fragestellung kann nun eine vorläufige Antwort gegeben werden: Die durchgeführte Unterrichtssituation hat sich einer für den Informatikunterricht nicht unbedingt klassischen Methodik bedient, die von den SuS als anregend und motivierend empfunden wurde. Als Lehrkraft konnte ich wahrnehmen, dass eine andere Art des Verständnisses des Lerngegenstandes auf Seiten der SuS gefördert werden konnte. Aus der in dieser Arbeit erfolgten Analyse und kritischen Betrachtung der Vorgehensweise anhand von didaktischer Prinzipien ist davon auszugehen, dass die Handlungsorientierung sehr wahrscheinlich zu einem Abbau von Lernhürden führen kann. Gleichzeitig ist nicht außer Acht zu lassen, dass immer noch die Vermittlung von Sprachkonzepten der ausgewählten Programmiersprache gefordert ist und somit nicht gänzlich auf sie verzichtet werden kann. Auch kann der Einsatz der erprobten Methodik bei einer ständigen Verwendung von den SuS leicht als ebenso anstrengend und unmotivierend empfunden werden, wie die Arbeit am Quellcode selbst. Um diese Problematik zu umgehen wäre folgendes Vorgehen denkbar: Für einen aus Kleinprojekten

aufgebauten Unterricht könnten unter einer bestimmten Fragestellung verschiedene Module angeboten werden. Diese würden für die SuS die Inhalte zum Selbststudium mit verschiedenen methodischen Umsetzungen anbieten. Aus den Modulen könnten die SuS auswählen und sich je nach Möglichkeit und Interesse das punktuell benötigte Wissen, oder auch ein breiteres Verständnis des jeweiligen Themas erarbeiten. Dies könnte mittels eines Programmierbeispiels oder einer handlungsorientierten Aufgabe geschehen.

Literaturverzeichnis

[Abt15] Dietmar Abts. *Grundkurs JAVA. Von Grundlagen bis zu Datenbank- und Netzwerkanwendungen*, 8., überarbeitete und erweiterte Auflage, Springer Vieweg, Wiesbaden, 2015

[Aeb11] Hans Aebli. *Zwölf Grundformen des Lehrens. Eine Allgemeine Didaktik auf psychologischer Grundlage. Medien und Inhalte didaktischer Kommunikation, der Lernzyklus*, 14. Auflage, Klett-Cotta, Stuttgart, 2011

[Alb] Claus Albowski. *Grafiksysteme am Beispiel eines Raumplaners*, http://www.informatik-hamburg.de/pmwiki.php/ObjektOrientierung/GrafiksystemeRaumplaner, Abgerufen am 04.11.2017

[Bar03] David J. Barnes, Michael Kölling. *Objektorientierte Programmierung mit Java. Eine praxisnahe Einführung mit BlueJ*, Übersetzt von Axel Schmolitzky, Pearson Studium, München, 2003

[Bre94] Norbert Breier. *Informatische Bildung als Teil der Allgemeinbildung*, LOG IN 14, H. 5/6., 1994

[Bzl16] Bärbel Barzel, Lars Holzäpfel, Timo Leuters, Christine Streit. *Scriptor Praxis: Mathematik unterrichten: Planen, durchführen, reflektieren*, 4. Auflage, Cornelsen Scriptor, Berlin, 2016

[Ehm09] Matthias Ehmann et al. *Duden Informatik - Sekundarstufe I / 9./10. Schuljahr - Objektorientierte Programmierung mit BlueJ*, Duden Schulbuchverlag Berlin Mannheim, 2009

[Ohm16] Fachschaft Informatik. *Schulinternes Curriculum Informatik. Sekundarstufe II Wahlbereich und Profile*, Hamburg, Stand: 14.03.2016

[HH09] Behörde für Schule und Berufsbildung Hamburg (Hrsg.). *Informatik – Bildungsplan Gymnasiale Oberstufe*, Hamburg, 2009

[HH11] Behörde für Schule und Berufsbildung Hamburg (Hrsg.). *Informatik Wahlfplichtfach – Bildungsplan Gymnasium Sekundarstufe I*, Hamburg, 2011

[HH14] Behörde für Schule und Berufsbildung Hamburg (Hrsg.). *Informatik Wahlpflichtfach – Bildungsplan Stadtteilschule Jahrgangsstufen 7 – 11*, Hamburg, 2014

[Hub07] Peter Hubwieser. *Didaktik der Informatik*, 3. Auflage, Springer-Verlag Berlin Heidelberg, 2007

[Hum02] Ludger Humbert, Sigrid Schubert. *Fachliche Orientierung des Informatikunterrichts in der Sekundarstufe II*, Didaktik der Informatik Universität Dortmund, Report Nr. 77, Februar 2002

[Koe93] Bernhard Koerber, Ingo-Rüdiger Peters. *Informatikunterricht und informationstechnische Grundbildung – ausgrenzen, abgrenzen oder integrieren?*, Troitzsch, S. 108–115, 1993

[Mey04] Hilbert Meyer. *Was ist guter Unterricht?*, Cornelsen Scriptor, Frankfurt/Main, 2004

[Mod11] Eckart Modrow. "Visuelle Programmierung – oder: Was lernt man aus Syntaxfehlern?", In: Marco Thomas (Hrsg.): *Informatik in Bildung und Beruf. 14. GI-Fachtagung „Informatik und Schule – INFOS 2011“*, S. 27–36, 2011

[Schwa07] Christine Schwarzer, Petra Buchwald. "Umlernen und Dazulernen.", In: Michael Göhlich, Christoph Wulf, Jörg Zirfas (Hrsg.): *Pädagogische Theorien des Lernens*, Beltz, Weinheim und Basel, S. 213–221, 2007

[Ull12] Christian Ullenboom. *Java ist auch eine Insel – Das umfassende Handbuch*, 10. Auflage, Galileo Press, Bonn, 2012

Anhang A

Schleifen

EINLEITUNG

Neben der Kontrollstruktur *Sequenz* gibt es noch die *Wiederholung*. Diese wird in Java durch *Schleifenkonstrukte* realisiert. Es gibt so genannte *Zählschleifen* und *bedingte Schleifen*.

ARBEITSAUFTRAG

1. Öffnet das Projekt *Iteration* und schaut euch die Klasse `Schleifendreher` an. Dort findet ihr Beispiele für die verschiedenen Schleifentypen in Java. Führt die Beispiele aus, um euch mit den Schleifen vertraut zu machen. Beachtet dabei die Ausgaben auf der Konsole.
2. Zeichnet ein Diagramm für den Ablauf von `for-`/`do-while-` und `while`-Schleifen und besprecht dieses mit eurem Sitznachbarn.
3. Schaut euch nun die Klasse Textanalyse des Projekts `Iteration` an. Dort gibt es eine vorgegebene Methode `istFrage(String text)`, die demonstriert, wie man die Länge eines Strings erhält und wie man auf einzelne Zeichen eines Strings zugreift. Probiert diese Methode interaktiv aus, indem ihr ein Exemplar von `TextAnalyse` erstellt und dann `istFrage` z.B. mit dem aktuellen Parameter „Wie geht's?“ aufruft.

   Worin unterscheiden sich die beiden Methoden `istFrage` und `istFrageKompakt`?
4. Schreibt nun eine eigene Methode `int zaehleVokale(String text)`, die für einen gegebenen Text als Ergebnis liefern soll, wie viele Vokale er enthält. Für den String „hallo“ soll die Methode beispielsweise eine 2 zurückgeben.

   Tipp: Verwendet in der Implementierung einen Schleifenzähler, der bei 0 beginnt und alle Positionen des Strings durchläuft. Erarbeitet euch ggf. die Funktionsweise einer `switch`-Anweisung, um die Aufgabe elegant zu lösen.

I

Ich versichere, dass ich diese Arbeit ohne fremde Hilfe verfasst und mich dabei anderer als der angegebenen Hilfsmittel nicht bedient habe. Die schriftliche Fassung entspricht der auf dem elektronischen Speichermedium.

Hamburg, Datum Unterschrift

Ich erkläre mich damit einverstanden, dass meine schriftliche Arbeit zur Zweiten Staatsprüfung der Lehrerbibliothek Hamburg für die Ausleihe zur Verfügung gestellt wird. Personenbezogene Daten sind derart anonymisiert, dass die Einzelangaben nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand einer bestimmten natürlichen Person zugeordnet werden können.

Hamburg, Datum Unterschrift