# Ein gamebasierter Ansatz zum Programmierunterricht in der Lehrerinnen- und Lehrerbildung

Nico Steinbach, Eckart Zitzler

**Abstract:** Mit dem Lehrplan 21 hält in den deutsch- und mehrsprachigen Kantonen der Schweiz der Themenbereich Informatik sowie die Kompetenzorientierung Einzug in die Volksschule. Dadurch ergeben sich neue Herausforderungen für die Schule, den Unterricht sowie die Aus- und Weiterbildung von Lehrpersonen. Zudem gibt es für diesen Themenbereich derzeit keine offiziellen Lehrmittel. Die pädagogischen Hochschulen stehen vor der Aufgabe, die aktiven Lehrpersonen, neben anderen Lehrplan-21-Weiterbildungen, für den Informatikunterricht zu befähigen.

In diesem Zusammenhang gilt es ein didaktisches Konzept zu entwerfen, welches den folgenden Anforderungen gerecht wird: 1) Einen motivierenden und spielerischen Zugang zu Informatikkonzepten zu ermöglichen 2) Die Heterogenität der Teilnehmenden hinsichtlich des Vorwissens und der Zielstufe zu berücksichtigen, 3) Die Erwachsenenbildung und den Schulunterricht gemeinsam zu denken, 4) Den Bezug zu anderen Fächern aufzuzeigen.

Der vorliegende Beitrag beschreibt die Entwicklung und Durchführung von kompetenzorientierten Weiterbildungen zum Themenfeld Programmieren im Kanton Bern. Im Zentrum des Ansatzes steht ein Jump'n Run-Computerspiel, welches blockbasiert entwickelt und individuell ausgestaltet werden kann.

**Keywords:** Computerspiel, Lehrerinnen- und Lehrerbildung, Programmierunterricht, Lehrplan 21

## 1 Einleitung

Mit dem Projekt Lehrplan 21 wurde in der Schweiz im Zeitraum 2010-2015 ein gemeinsamer Lehrplan für die 21 deutsch- und mehrsprachigen Kantone entwickelt mit der Absicht, die Ziele der Volksschule zu harmonisieren. Eine wesentliche Neuerung ist, dass die Inhalte kompetenzorientiert formuliert und auf Zyklen verteilt sind (Zyklus 1 = Kindergarten bis zweite Klasse; Zyklus 2 = dritte bis sechste Klasse; Zyklus 3 = siebte bis neunte Klasse). Des Weiteren wurde erstmalig die Informatik in allen Zyklen verankert, wobei auch die Bezüge zu anderen Fächern thematisiert werden. Sie umfasst neben reinen Anwendungskompetenzen drei Kompetenzen zu den Themenfeldern Datenstrukturen, Algorithmen und Informatiksystemen, die jeweils in mehreren Kompetenzstufen ausdifferenziert sind. Je nach Kanton sind separate Zeitgefäße hierfür vorgesehen, die zudem vier Kompetenzen im Medienbereich umfassen. Im Kanton Bern handelt es sich um jeweils eine Wochenlektion im 5., 6., 7. und 9. Schuljahr, deren Umsetzung im Schuljahr 2018/19 beginnt.

Da derzeit keine entsprechenden Informatik-Lehrmittel zur Verfügung stehen und den meisten im Berufsfeld tätigen Lehrpersonen sowohl die fachlichen als auch die

fachdidaktischen Hintergründe fehlen, stehen die Schulen vor großen Herausforderungen. Für die Pädagogischen Hochschulen heißt das, eine große Anzahl von Lehrerinnen und Lehrern in kurzer Zeit – im Kanton Bern sind es gut 750 Personen – innerhalb von drei Jahren für den Informatikunterricht vorzubereiten, neben den anderen Weiterbildungen im Kontext des Lehrplan 21. Dabei ergeben sich diverse Fragen, die sich bei der Entwicklung von Weiterbildungsangeboten stellen: 1. Wie kann der Spanne zwischen keinem Vorwissen und bereits eigener Unterrichtserfahrung begegnet werden?; 2. Wie kann das zu erarbeitende Konzept inklusive Material aufbereitet werden, damit es sowohl für die Lehrerinnen- und Lehrerbildung als auch für den Unterricht genutzt werden kann?; 3. Wie kann der Bezug zu anderen Fächern aufgezeigt und implementiert werden?

An der Pädagogischen Hochschule Bern wurde ein einwöchiger Blockkurs für Volksschullehrpersonen entwickelt, in dem sowohl die Medien- als auch die Informatikkompetenzen behandelt werden. Dieser Beitrag fokussiert die Algorithmik und diskutiert, wie das Themenfeld der Programmierung innerhalb eines Tages in diesen Blockkurs eingebettet wurde. Im Folgenden wird die Entwicklung eines entsprechenden didaktischen Konzepts vorgestellt und die Umsetzung im Schulunterricht sowie in der Weiterbildung diskutiert.

### 1.1 Hintergrund

Im Abschlussbericht der Arbeitsgruppe ICT und Medien, die das Modul Medien und Informatik des Lehrplan 21 ausgearbeitet hat, wird für die Weiterbildung von amtierenden Lehrpersonen ein Umfang von mindestens 3 ECTS-Punkten (1 ECTS-Punkt entspricht 30 Stunden) für Primarlehrpersonen bzw. 5 ECTS-Punkten für Sekundarlehrpersonen empfohlen [MI15]. Aufgrund verschiedener praktischer Rahmenbedingungen steht den Lehrpersonen jedoch in der Regel wesentlich weniger Zeit zur Verfügung, um sich im Themenfeld aus- und weiterzubilden.

### 1.2 Rahmenbedingungen im Kanton Bern

Im Kanton Bern ist die Weiterbildung zum Modul Medien und Informatik nicht obligatorisch. Es liegt in der Verantwortung der Schulleitungen zu bestimmen, welche Lehrpersonen die spezifischen Lektionen in der 5., 6., 7. und 9. Klasse unterrichten werden. Weil zudem eine Vielzahl weiterer, obligatorischer Weiterbildungen im Kontext des Lehrplan 21 absolviert werden müssen, die mit ca. 12 Stunden pro Fach dotiert sind, sind die zeitlichen Ressourcen für das Themenfeld Medien und Informatik begrenzt. In Absprache mit der Erziehungsdirektion wurde ein 30-stündiges, einwöchiges Weiterbildungsformat entwickelt, wobei drei Tage den Medien-Kompetenzen und zwei Tage den Informatik-Kompetenzen gewidmet sind. In diesen Blockwochen mit knapp 50 Teilnehmenden sind sowohl Primarlehrpersonen (betrifft 5. und 6. Klasse) als auch Sekundarlehrpersonen (betrifft 7. und 9. Klasse) weiterzubilden. Aufgrund der

beschränkten personellen Ressourcen sind die Teilnehmendengruppen gemischt, zudem bringen die Teilnehmenden sehr unterschiedliche persönliche Erfahrungen zum Thema Medien und Informatik mit.

Aufgrund dieser Rahmenbedingungen ist die Zielsetzung des Blockkurses vor allem den Lehrpersonen die fachlichen Hintergründe zum Modul Medien und Informatik zu vermitteln, fachdidaktische Ideen aufzuzeigen und Motivation und Interesse für das Thema zu wecken, damit diese sich selbständig weiter mit dem Thema beschäftigen können. Insbesondere ergeben sich folgende Anforderungen an das Kurskonzept: 1) Es kann Teilnehmende unterschiedlichen Vorwissens abholen und ins Thema einführen beziehungsweise weiterführende Themen anbieten; 2) Es ist geeignet für die Lehrerinnen- und Lehrerbildung als auch den anschließenden Einsatz im Unterricht; 3) Es zeigt konkrete Verknüpfungen zu anderen Fachbereichen auf; 4) Es spricht die Kreativität an und motiviert durch Lebensweltbezug.

Die Entwicklung solcher Materialien und Konzepte obliegt normalerweise den kantonalen Lehrmittelverlagen und Lehrmittelautoren. Eine Weiterbildung kann sich dann auf die vorgeschriebenen Lehrmittel beziehen. Zum Zeitpunkt der ersten Durchführung des Blockkurses lag jedoch kein entsprechendes Lehrmittel vor. Aus diesem Grund wurde die Entwicklung eines eigenen Konzepts angestrebt. Im Folgenden fokussieren wir uns auf den Aspekt des Programmierunterrichts – ein Tag im Blockkurs –, der im Lehrplan unter der Kompetenz Algorithmen beschrieben wird: “Die Schülerinnen und Schüler können einfache Problemstellungen analysieren, mögliche Lösungsverfahren beschreiben und in Programmen umsetzen.” [LP16] Die Kompetenz unterteilt sich in neun Kompetenzstufen, die mehrheitlich im zweiten Zyklus liegen. Die Frage, mit der sich dieser Beitrag auseinandersetzt, lautet, wie eine eintägige Weiterbildung zum Thema Programmierung didaktisch konzipiert und umgesetzt werden kann, sodass die oben aufgeführten Anforderungen erfüllt werden können.

### 1.3 Einbettung

Es existiert eine Vielzahl von Ansätzen und entsprechender Erfahrung zum Programmierunterricht in der Schule. Neben dem Programmieren ohne Computer können vor allem die textbasierte und die visuelle Programmierung genannt werden. Die textbasierte Programmierung blickt auf eine langjährige Debatte nach der Wahl einer geeigneten Programmiersprache, nebst Programmierparadigma zurück. Das Schreiben von textbasierten Programmen kann Schülerinnen und Schülern Schwierigkeiten bereiten, obwohl die Motivation und Fähigkeit zum algorithmischen Problemlösen vorhanden ist [MMS11]. Eine mögliche Konsequenz ist, dass das Interesse an der Informatik sinkt [Hu00]. Als Argumente gegen die textbasierte Programmierung werden unter anderem aufgeführt: Die Kodierung des Programmcodes in Textform ist nicht intuitiv [MMS11] und Menschen können Bilder viel früher verstehen als Text [Sc01]. Für die Förderung algorithmischen Denkens bietet die visuelle Programmierung für Anfänger Vorteile. Sie erlaubt es, sich schneller auf den Inhalt zu fokussieren und

vermeidet Syntaxfehler [Mo11].

Mit Scratch – die offizielle erste Version wurde 2007 veröffentlicht – existiert eine solche visuelle Programmierlernumgebung, die sich direkt an Kinder und Jugendliche richtet, blockbasierte Entwicklung von Programmen ermöglicht und Spiele ins Zentrum stellt. Neben Weiterentwicklungen (Scratch 2.0, 2013) und Modifikationen (BYOB/Snap) gibt es auch weitere Plattformen wie code.org [CO13], die auf die Kombination aus Blöcken und Spielen setzen. Mit der Bewegung Hour-of-Code [HO13] und prominenter Unterstützung (u.a. Barack Obama, Mark Zuckerberg, Bill Gates) schaffte es die Programmierung in die US-Schulen.

Der Einsatz von Spielen, wie er auch bei code.org verfolgt wird, scheint vielversprechend. Allein bis Ende 2015 konnte code.org über 100 Millionen Schülerinnen und Schüler auf der ganzen Welt erreichen [US15]. Auch die Bitkom konnte aufzeigen, dass sich Gaming in allen Altersgruppen etabliert hat [BI15]. Spiele sind dankbare Vehikel für Lerninhalte [Pr01]. Allerdings sind die zugrundeliegenden didaktischen Ansätze nur begrenzt auf das oben skizzierte Weiterbildungsszenario übertragbar. Die Aufgaben bei code.org schränken die Lernenden ein, indem sie die zu nutzenden Blöcke und damit das Wissen zur Lösung einer Aufgabe vorgeben. Schon Romeike kritisierte, dass Problemstellungen (Start, Ziel, Hürde) in der Informatik eher als Aufgaben (vorgegebenes Wissen und definierte Lösung) formuliert werden. Die Kreativität die der Informatik immanent ist, kommt nicht zur Entfaltung [Ro08]. Zusätzlich erschwert ein solches Setting die Fächerverbindung, da es kaum Ansatzpunkte gibt, Inhalte anderer Fächer einzubinden.

## 2 Didaktischer Ansatz

In diesem Beitrag wird die Idee verfolgt, die Lernenden ein Computerspiel (weiter)entwickeln zu lassen und so die Kompetenzstufen im Bereich der Algorithmik abzudecken. Beim Computerspiel handelt es sich um ein einfaches Jump'n-Run-Computerspiel, in dem eine Spielfigur von links nach rechts durch Spielumgebungen geführt werden kann. Eine Grundfunktionalität soll zur Verfügung gestellt werden, die dann von den Lernenden schrittweise erweitert wird, z. B. indem die Spielfigur neue Fähigkeiten erhält oder die Spielumgebung ausgebaut wird.

**Konzept.** Das didaktische Konzept, das an der PHBern entwickelt wurde, basiert auf drei Eckpfeilern:

1. Eine blockbasierte Programmierumgebung, die einen möglichst einfachen Zugang zum Programmieren ermöglicht, den Austausch mit anderen Programmierenden fördert, frei verfügbar ist und zudem erlaubt, Multimedia-Inhalte niederschwellig einzubinden; Letzteres war uns wichtig, um die inhaltliche Anbindung an andere Fächer zu erleichtern.

2. Ein erweiterbares Programmgerüst, welches die Grundfunktionalität des Computerspiels implementiert, gleichzeitig aber möglichst schlank gehalten ist, damit die Lernenden das Spiel erweitern und abändern können. Der Nachteil eines solchen vorgegebenen Systems ist, dass die Lernenden sich überfordert und verloren fühlen können. Für uns stand jedoch aufgrund der Heterogenität der Lerngruppen im Vordergrund, dass früh sichtbare Erfolgserlebnisse und zudem sehr offene Aufgabenstellungen möglich sind.
3. Eine interaktive Wegleitung, anhand derer die Lernenden das Computerspiel individuell weiterentwickeln können. Sie besteht aus mehreren multimedialen Aufgabenblättern Jedes Aufgabenblatt enthält eine Sequenz von Aufgaben, die durch Zusatzinformationen (Videos, Texte etc.) und Beispiele angereichert sind. Die insgesamt neun Aufgabenblätter decken alle Kompetenzstufen im Bereich Algorithmik ab.

Wir haben uns bewusst dafür entschieden, die Lernenden mit einer vollständigen Programmierumgebung zu konfrontieren und die Möglichkeiten der Programmierung in Abhängigkeit der Aufgabenstellung explizit nicht einzuschränken. Des Weiteren folgt die Konzeption der Aufgabenblätter der Idee der Lernspirale [Ki13]: Anstatt fundamentale Programmierkonzepte einzeln zu behandeln, werden Konstrukte wie Anweisungen, bedingte Anweisungen, Schleifen und Variablen in allen Aufgabenblättern benutzt. Die Lernenden begegnen den elementaren Ideen also immer wieder, allerdings nimmt die Komplexität der Aufgaben zu. Damit einher geht auch das Konzept, dass die Aufgabenblätter nicht linear, sondern eher baumartig angeordnet sind. Einzig die zwei ersten Aufgabenblätter sind hintereinander zu bearbeiten, danach können die Lernenden nach ihren Fähigkeiten und Interessen auswählen.

**Implementierung.** Als Programmierumgebung wurde Scratch [SC17] gewählt. Scratch erfüllt die oben beschriebenen Anforderungen, hat allerdings (derzeit) den Nachteil, dass die Umgebung auf Geräten ohne Flash nicht einsetzbar ist; diese Einschränkung soll gemäß Ankündigung der Scratch-Entwickler behoben werden [SC16]. Das Programmgerüst, um die Grundfunktionalität des Spiels zu realisieren, umfasst fünf Funktionen und insgesamt knapp vierzig Einzelanweisungen; es definiert die Spielfigur sowie deren Bewegung über die Pfeiltasten und wird in Abbildung 1 skizziert.

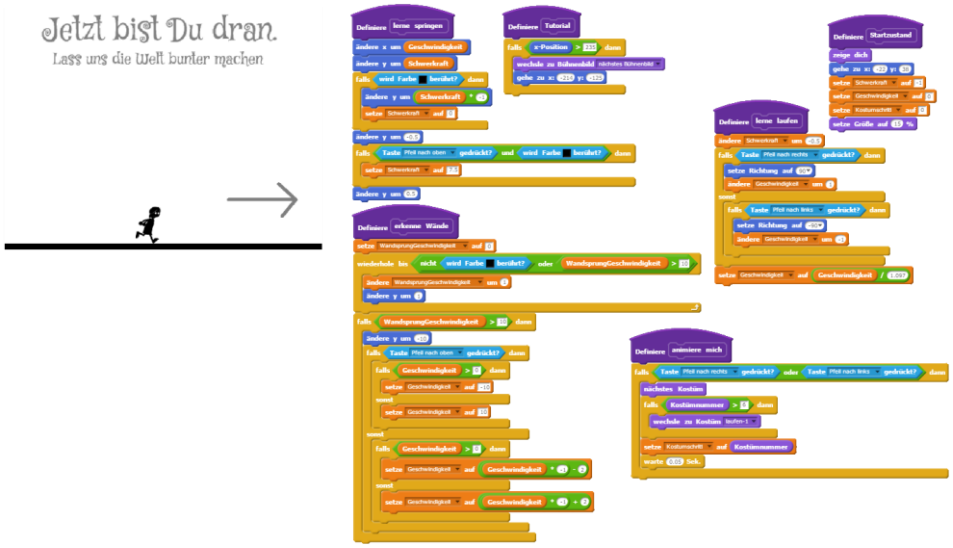


Abbildung 1: Das Spielfeld (links) und das Programmgrundgerüst in Scratch (rechts)

Die Arbeitsumgebung umfasst drei Einstiegshilfen in Scratch, neun Arbeitsblätter sowie einen Remixbaum (alle Weiterentwicklungen unseres Ursprungprojektes, die öffentlich sind), die in Form einer Baumstruktur auf einer Webseite verlinkt sind. Die vollständigen Materialien finden sich unter https://milehrplan.ch/scratch.

Die Arbeitsblätter werden on-the-fly aus einem JSON-Format generiert, welches in einer relationalen Datenbank liegt. Das ermöglicht sowohl eine Ansicht für Schülerinnen und Schüler ohne Lehrerkommentare, als auch eine weitere Ansicht für Lehrpersonen, zu erzeugen. Die Arbeitsblätter setzen sich wie folgt zusammen: Zwei fundamentale Arbeitsblätter (*"Der Beginn", "Lass uns die Welt bunter machen"*) mit denen begonnen wird. Sie greifen niederschwellig einfache Anweisungen, bedingte Anweisungen sowie Schleifen auf und dienen dazu, der Spielfigur erste Interaktionen mit ihrer Umgebung zu ermöglichen, indem sie auf Farben reagiert. Aus den restlichen sieben Arbeitsblättern kann frei gewählt werden. Die drei fachübergreifenden Arbeitsblätter (*"Level Design", "Story Design", "Sound Design"*) dienen zur Erzeugung von Spielwelten (Bildnerisches Gestalten), Geschichten und Anleitungen (Deutsch) sowie Hintergrundmusik und Soundeffekten (Musik). Das erzeugte Material kann über bedingte Anweisungen in das Computerspiel integriert werden. Vier weitere Arbeitsblätter routinieren die Verwendung der eingeführten Konzepte und führen neue ein. Es handelt sich um Variablen (*"Spielelement - Sammeln"*), Nachrichten zwischen Objekten (*"Spielelement - Hindernis"*), Ereignisse (*"Spielelement - Aktion"*) sowie die Implementierung weiterer Objekte wie zum Beispiel Gegner mit zufallsbasierten Bewegungspfaden (*"Spielelement - Gegner"*).

Abbildung 2 stellt zwei Programmblöcke gegenüber – von einer früheren und einer

späteren Aufgabe – und vermittelt einen Eindruck vom Anforderungsniveau und der Komplexitätszunahme. Im linken Programmblock wird der Figur beigebracht, auf verschiedene Farben zu reagieren und entsprechend einfache Anweisungen auszuführen. Die Schleife simuliert den Gameloop, der nötig ist, um Interaktionen bis zum Programmende am Leben zu erhalten. Im rechten Programmblock wird entschieden, was mit einem zusätzlichen Objekt (Gegner) passiert, wenn die Variable *"LebenVirus"* auf unter 1 fällt. Dieser Programmblock ist nur ein Teil einer möglichen Lösung und interagiert mit weiteren Blöcken, welche die Variable ändern dürfen und Nachrichten schicken (nicht im Bild enthalten).

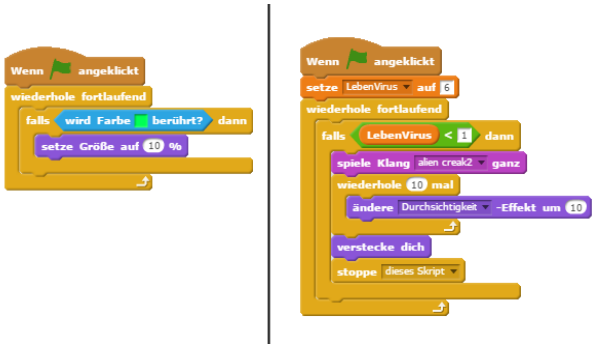


Abbildung 2: Gegenüberstellung zweier Programmblöcke unterschiedlichen Niveaus

## 3 Erfahrungen

Das vorgestellte Konzept wurde einerseits in Schulklassen getestet und andererseits in der Weiterbildung eingesetzt.

Der Einsatz im Unterricht fand in vier Klassen des siebten Schuljahres einer Sekundarschule statt mit je 16-18 Schülerinnen und Schülern. Die Erprobung dauerte über sechs Wochen mit je einer Lektion pro Woche. Stellvertretend für die Klassenlehrperson übernahm ein Mitarbeiter der PHBern die Rolle der Lehrperson, zwei weitere Mitarbeiter waren als Beobachter im Klassenraum. Das Ziel der sechs Wochen war die fachübergreifenden Themen zu erreichen, da die gesamte Einheit für bis zu 16 Lektionen ausgelegt ist. Im Vorfeld zu jeder Lektion entschied die Lehrperson, welche Arbeitsblätter zur Verfügung stehen. Anschließend haben sich die Schülerinnen und Schüler, meist zu zweit, selbstständig an die Inhalte gewagt.

In der allgemeinen Beobachtung und persönlichen Gesprächen fiel auf, dass das Computerspiel als Thema von beiden Geschlechtern sehr gut angenommen wurde. Die Arbeitsblätter wurden in unterschiedlichem Tempo im Pair Programming bearbeitet. Vor allem schnelle Teams halfen den anderen mit Hinweisen. Viele Teams haben intuitiv begonnen, aus den Problemstellungen eigene Projekte zu definieren. Alle Teams hatten

völlig individuelle Ergebnisse, durchliefen aber dieselben Konzepte. Auch über die Unterrichtszeit hinaus gab es zusätzliches Engagement in den Pausen und in der Freizeit. Die Lehrperson hat bei Fragen vor allem zwischen den Teams vermittelt und Konzepte im Plenum zusammengefasst. Am Ende der sechs Wochen wurden die entstandenen Projekte auf Initiative der Schülerinnen und Schüler abschließend vorgestellt.

Einschränkend war der 45-Minuten-Rahmen der pro Woche zur Verfügung stand. In der Regel benötigten die Schülerinnen und Schüler das erste Unterrichtsdrittel, um am Stand der letzten Woche anzuknüpfen. Ebenfalls kann es für die Lehrperson mitunter schwierig sein, an die unterschiedlichen Projekte und Lernstände anzuknüpfen nach einer Woche Pause zwischen den Lektionen.

In der Lehrerinnen- und Lehrerbildung haben bisher 95 Lehrpersonen die Blockwoche absolviert. An der ersten Blockwoche nahmen 48 Lehrpersonen teil, an der zweiten 47. Die Teilnehmenden wurden jeweils auf zwei Räume verteilt sowie zwei Dozenten zugeteilt. Die Kurstage erstreckten sich von 8:30 Uhr bis 17:00 Uhr. Zu Beginn des Algorithmentages wurden die Teilnehmenden in den Räumen in Vierer-Gruppen aufgeteilt. Anschließend wurde die Relevanz der Informatik für die Gesellschaft und Schule thematisiert und im Plenum die Vorerfahrung der Teilnehmenden diskutiert, bevor die selbstständige Arbeit begann.

Positiv hat sich gezeigt, dass die Thematik Computerspiele entgegen genereller Vermutungen auch bei Lehrpersonen Anklang findet. Zum einen, weil ihre Schülerinnen und Schüler oft über Games erzählen und sie motiviert sind, ihnen das Material vorzustellen. Zum anderen, weil sie Scratch und das Material als kreativ wahrnehmen und aktiv gestalten - im Gegensatz zu ihrer Vorstellung vom reinen Spielen kommerzieller Computerspiele. Selbst Lehrpersonen, die noch nie programmiert haben und/oder noch nie gespielt haben, nahmen den Ansatz dankend auf. Wie auch bei den Schülerinnen und Schülern gab es bei den Teilnehmenden unterschiedliche Arbeitstempi. Sie arbeiteten ebenfalls im Pair Programming und fingen an, eigene Projekte und kreative Umsetzungen aus den Arbeitsblättern zu generieren. Alle Teilnehmenden erreichten gegen Mittag die fächerübergreifenden Themen und nutzten den Nachmittag, neue Konzepte in den anderen Arbeitsblättern zu streifen.

Kritisch muss gesagt werden, dass an einem Kurstag keine echte Algorithmik erlernt werden kann und die Teilnehmenden die meisten Konzepte nur ausprobieren können. Die meisten reflektieren das für sich und geben an, sich weiterhin nicht sicher zu fühlen, das Thema im Unterricht aufzugreifen. Für sie bildet das Konzept nebst Material einen soliden Ausgangspunkt.

In einer abschließenden Befragung zu der gesamten Blockwoche wurde der Kenntnisstand in der Informatik vor und nach der Woche erfragt, siehe Abbildung 3.

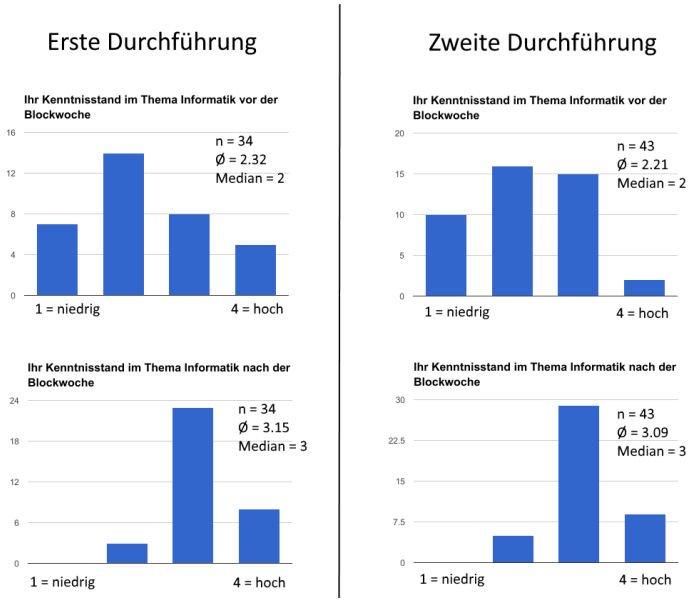


Abbildung 3: Kenntnisstand zur Informatik vor und nach den Blockwochen (Selbsteinschätzung)

Die in Abbildung 3 gezeigten Resultate beziehen sich auf die beiden Informatiktage der Woche zu den Themen Algorithmen, Datenstrukturen und Informatiksysteme. Es zeigte sich nach der Blockwoche, dass sowohl ein deutlicher Lernzuwachs zu verzeichnen ist, als auch dass sich keine Lehrperson nach der Woche selbst mit einem niedrigen Kenntnisstand einstuft.

## 4 Fazit

Das vorliegende didaktische Konzept ist geeignet um mit heterogenen Gruppen bestehend aus Lehrpersonen in der Lehrerinnen- und Lehrerbildung oder Schülerinnen und Schülern im Unterricht zu arbeiten. Ebenso vermag die Thematik eines Computerspiels beide Gruppen gleichermaßen anzusprechen und bietet in Verbindung mit einer visuellen Programmierlernumgebung wie Scratch eine niedrige Einstiegshürde für Programmieranfänger. Zudem eignet sich der Kontext rund um Spiele besonders gut, um kompetenzorientierte Problemstellungen zu formulieren, verschiedene Lösungswege zuzulassen und Raum für kreative Weiterentwicklungen zu ermöglichen. Das Konzept stellt hohe Anforderungen an Dozierende und Lehrpersonen und setzt voraus, dass sich Lernende gegenseitig unterstützen können.

Offen bleibt die Frage, wie nachhaltig eine eintägige Einführung in die Algorithmik sein

kann, schließlich lässt sich an einem Tag nur ein Einblick geben. Die Lehrpersonen gaben an, dass sie sich nach dem Tag nicht sicher fühlen, das Thema zu unterrichten, mit dem Material und der exemplarischen Durchführung jedoch eine solide Basis zum Anfangen gegeben ist. Die selbstständige Auseinandersetzung sowie weitergehende Weiterbildungsangebote sind eine Voraussetzung, damit das Thema effektiv in den Schulen unterrichtet werden kann. Erst dann kann sich ein didaktischer Diskurs bilden.

## Literaturverzeichnis

[BI15] Gaming hat sich in allen Altersgruppen etabliert, https://www.bitkom.org/Presse/Presseinformation/Gaming-hat-sich-in-allen-Altersgruppen-etabliert.html, Stand:14.02.2017

[CO13] Code.org - Jeder Schüler sollte die Möglichkeit haben, Informatik zu lernen, http://code.org, Stand: 14.02.2017

[Hu00] Bericht zur Lehrerausbildung Informatik, http://koenigstein.inf.tu-dresden.de/00/humbert2.html, Stand: 14.02.2017

[HO13] Hour of Code, https://hourofcode.com/de, Stand: 14.02.2017

[Ki13] Klippert, Heinz, Leitfaden zum Arbeiten mit Lernspiralen, 2013

[LP16] Kompetenz Algorithmen im Modullehrplan Medien und Informatik, http://v-ef.lehrplan.ch/index.php?code=a|10|0|2|0|2, Stand: 14.02.2017

[MI15] Schlussbericht der Arbeitsgruppe zu Medien und Informatik im Lehrplan 21, https://www.lehrplan.ch/sites/default/files/Schlussbericht_MI_2015-02-23%20mit%20Anhang_0.pdf, Stand: 14.02.2017

[MMS11] Modrow Eckart, Jens Mönig, Kerstin Strecker, LOG IN Heft Nr. 168, 2011

[Mo11] Modrow, Eckart, Visuelle Programmierung: - oder: Was lernt man aus Syntaxfehlern?, Lecture Notes in Informatics, Proceedings, Volume P-249, 2011

[Pr01] Prensky, Marc, Digital Game- Based Learning, McGraw-Hill, 2001

[Ro08] Romeike, Ralf, Kreativität im Informatikunterricht, Mathematisch-Naturwissenschaftliche Fakultät Potsdam, 2008

[SC17] Scratch - imagine, program, share - https://scratch.mit.edu, Stand: 14.02.2017

[Sc11] Schiffer Stefan, Visuelle Programmierung, Grundlagen und Einsatzmöglichkeiten, 2001

[SC16] Scratch-Blöcke (Google Blocky und Scratch - Open Source Projekt), https://scratch.mit.edu/developers, Stand: 14.02.2017

[US15] Hour of Code to feature Star Wars: The Force Awakens. In: USA TODAY ,9. November 2015