Fernlehren und Fernlernen von Objektorientierter Programmierung (OOP)

Mathias Ellmann¹

Online publiziert: 13. November 2020
© Der/die Autor(en) 2020

Zusammenfassung

Dieses Papier beschreibt unsere Lehrerfahrungen in einer virtuellen Lehr- und Lernumgebung für eine Vorlesung über objektorientierte Programmierung (OOP) in Java. Wir verwenden die Live-Meeting-Anwendung Adobe Connect sowie Lern- und Lehrmethoden für OOP-Vorlesungen wie Online-Coding, Online-Debugging, Online-Gruppenräume, Online-Whiteboards und Online-Fragebögen, um unsere Lehr- und Lernziele zu messen und zu erreichen. Wir haben festgestellt, dass Online-Coding in Kombination mit Online-Debugging oder einem Codebeispiel das mit einem Kommentar versehen ist, in dieser virtuellen Lehr- und Lernumgebung am effektivsten funktioniert, um den Studierenden ein besseres Verständnis der Programmierkonzepte und -methoden zu vermitteln. Online-Coding und Online-Debugging sollten unmittelbar nach der Vermittlung der OOP-Konzepte und -Methoden durchgeführt werden. Andere Lehrmethoden wie das Unterrichten der OOP-Konzepte ohne ein konkretes Beispiel oder eine Anwendung führen zu Frustration und Unzufriedenheit bei den Studierenden.

Einleitung

Heutzutage wird der Unterricht mehr und mehr online angeboten [1, 2]. Es ist möglich, einen kompletten Master- oder Bachelor-Studiengang in einem MOOC (Massive Open Online Course) [3] wie in coursera¹ oder MOODLE [4, 5] durchzuführen. Studien hatten eine hohe Abwanderungsrate von MOOCs gezeigt, z. B. wegen der fehlenden Gelegenheiten, online mit einem Dozenten oder anderen Studierenden zu sprechen [6]. Infolgedessen fühlten sich die Studierenden beim Studieren und Zuhören des Dozenten isoliert. Eine andere Möglichkeit, online in einer virtuellen Umgebung zu unterrichten, ist ein Videokonferenzsystem für E-Learning-Zwecke, in das sich die Studierenden integriert und nicht isoliert fühlen [6]. Ein Beispiel für ein solches E-Learning-System ist Adobe Connect², in dem eine Online-Diskussion mit einem Dozenten und Gruppenräumen möglich ist. Diese Umgebung bietet viele Möglichkeiten,

online zu lehren und die Vorlesungen einzurichten, indem sie die effektivsten Phasen (z. B. Üben durch Ausprobieren, andere unterrichten oder Gruppendiskussionen [7]) in der typischen Lernpyramide [7, 8] abdeckt und den Studierenden Feedback gibt, um bessere Lernergebnisse zu erzielen [9, 10].

Missverständnisse in Bezug auf das Verständnis der OOP-Terminologie und -Konzepte sind ein bekanntes Problem [11]. Mit den Möglichkeiten, die Adobe Connect bietet, um die Studierenden in den Lernprozess einzubeziehen und ihnen Feedback [9, 10, 12] im gleichen Lehr- und Lernkontext zu geben, wollen wir die Missverständnisse vermeiden und ihre Bedürfnisse im Lernprozess [9] während der Vorlesung in der OOP abdecken. Zudem können Sie sich durch das Lernen im Kontext des zukünftigen Arbeitskontextes – der virtuellen Umgebung – leichter an die Inhalte zu OOP erinnern [13], da Sie sich wieder im demselben Kontext wie im Lernkontext befinden [2, 13].

Viele Studierende beschreiten Ihr Studium heutzutage nebenberuflich [14]. Ein effektives Anwenden der heute bekannten Lehr- und Lernmethoden [15] ist heute umso wichtiger, sodass die Studierenden, weniger belastend die komplexen Inhalte insbesondere im Objektorientierten Programmieren erlernen können und diese im Arbeitskontext handlungsfähig sind.

¹ <https://de.coursera.org>.

² <https://adobeconnect.eu>.

✉ Mathias Ellmann
mathias.ellmann.cs@gmail.com

¹ DIPLOMA University of Applied Sciences, Bad Sooden-Allendorf, Deutschland

Kurs-Aufbau

Einzelheiten zum Kurs

Die Modulvorlesung in objektorientierter Programmierung (OOP) wird in einem zweisemestrigen Kurs an der DIPLOMA Fachhochschule im Studiengang Wirtschaftsinformatik gehalten. In einem Semester wird der Kurs Grundlegende Programmieretechniken und im nachhergehenden Semester wird der Kurs Objektorientiertes Programmieren gehalten.

Im Kurs Grundlegende Programmieretechniken lernen die Studierenden grundlegende Methoden der objektorientierten Programmierung in Java kennen. Die Lehrveranstaltung behandelt die Grundelemente von Java wie Kontrollstrukturen, Klassen und Vererbung sowie Algorithmen wie Bubble Sort. Im Kurs Objektorientierte Programmierung werden ausgewählte API-Elemente in Java wie Netzwerkverbindungen, Sockets oder UI-Programmierung mit dem Abstract Window Toolkit (awt) behandelt. Zu jedem Kurs gibt es ein Kursbuch von ca. 80 Seiten mit Beispielen und Übungen, die speziell für das Selbststudium der objektorientierten Programmierung geschrieben sind.

In der Vorlesung verteilen wir Vortragsfolien, die ein OOP-Thema aus dem Kursbuch zusammenfassen und Online-Übungen zu einem Thema als Kontrollstrukturen in OOP enthielten. Wir diskutierten jedes Thema unter Verwendung von Adobe Connect und seinen Lehr- und Lernfunktionen. Die Präsenzzeit jeder Vorlesung beträgt etwa 20 Stunden (insgesamt für das OOP-Modul 40 Stunden), so dass die Online-Vorlesung das Selbststudium der Studierenden erweitert. In der Regel treffen sich die Studierenden mit dem Dozenten am Samstag von 09:30–12:45 Uhr oder nachmittags von 13:15–16:30 Uhr (in der Regel in einem Block von insgesamt 4–5 Wochenenden nacheinander).

Durch die Kombination von Studienbüchern und der virtuellen Lehr- und Lernumgebung können alle Aspekte der

Bloom-Taxonomie [16] abgedeckt werden, wie Tab. 2 zeigt. Das Studienbuch enthält Übungen, die die Studierenden innerhalb des Semesters lösen können. Die Dozentin oder der Dozent stellt Übungen in den Gruppenräumen zur Verfügung. Die Studierenden können ihre Aufgabe am Whiteboard lösen und die Lösungen sowohl online mit der Gruppe als auch mit dem Dozenten diskutieren. Die Studierenden können auch ihre Lösungen aus den Übungen im Studienbuch mit der Lösung der Übung vergleichen und analysieren. Die Studenten können die Lösungen in der Live-Coding, in der Online-Debugging-Sitzung sowie in den Code-Beispielen analysieren. Die Studierenden können das Wissen über OOP in ihrer IDE oder in der virtuellen Umgebung in den Gruppenräumen, in der Live-Codingsitzung und am Whiteboard anwenden. Die Studierenden führen einen Selbsttest durch, in dem sie ihr Verständnis und ihre Erinnerung an den OOP-Inhalt im Online-Fragebogen unter Beweis stellen.

Studenten

Die Lerngruppe besteht in der Regel aus 6–10 Studierenden. Die Studierenden haben seit 1–2 Jahren Erfahrungen mit Informatik-Themen (sie haben bereits die Programmiersprache C++ gelernt). Die Studierenden haben in der Regel einen starken technischen Hintergrund, aber nicht konsequenterweise in Informatik als im Maschinenbau als Techniker. Die Studierenden studieren samstags in Teilzeit und nebenberuflich nach einer 35–40-Stunden-Woche. Alle Vorlesungen werden online in einer virtuellen Lehr- und Lernumgebung gehalten, in der die Studierenden mit ihren Videokameras und Mikrofonen anwesend sind. Die Studierenden verbinden sich in der Regel zu Hause mit dem Dozenten.

Infrastruktur

Für diesen Kurs verwenden wir die Live-Meeting-Anwendung Adobe Connect (siehe Abb. 1 und Abb. 2). Die Lehr- und Lernfunktionen in Adobe Connect werden als pods [17, 18] bezeichnet. Wir haben die Vorlesungsfolien online mit der File-Sharing-Funktion geteilt und während der Vorlesung mehrere Pods verwendet. Wir beschreiben kurz die Pods in Tab. 1.

Fernlehren und Fernlernen

Im Folgenden beschreiben wir unsere Erfahrungen nach vier Semestern Vorlesungszeit der Modulvorlesung OOP. Wir haben die Modulvorlesung Objektorientierter Programmierung vor acht Studentengruppen (zu jeweils 4 Gruppen in Grundlegende Programmieretechniken und Objektorien-

Tab. 1 Verwendete Pods (Lehr- und Lernfunktionen) in der Adobe Connect-Anwendung

Pod	Beschreibung und Verwendung
Filesharing	Teilen und Besprechen von Folien eines OOP-Themas
Whiteboard	Zeichnen z. B. UML-Modelle online
Bildschirm freigeben	Gemeinsamer Bildschirm für Online-Coding und Online-Debugging
Video des Dozenten	Virtuelle Kommunikation mit anderen Studierenden
Gruppenraum	Zusammenarbeit mit Studierenden
Chat für Fragen	Diskussion mit Studierenden
Online-Fragen	Online-Fragebogen, um z. B. den Lernfortschritt zu messen
Weblink	Studenten in ihrem Browser zu einer API-Referenzdokumentation führen

Abb. 1 Live-Meeting-Anwendung mit Adobe Connect

Tab. 2 Deckung der Bloom's Pyramide (neue Version) [16] mit Studienbuch und virtuellem Lehr- und Lernkontext [2]

Stage	Study book	Virtueller Lehr- und Lernraum
Creating	In-Buch-Übungen	Gruppenräume und Whiteboard
Evaluating	In-Buch-Lösungen	Online-Diskussionen
Analyzing	In-Buch-Lösungen	Live-Coding, Online-Debugging, Code-Beispiele
Applying	Eigene IDE	Gruppenräume, Live-Coding und Whiteboard
Understanding	Selbsttest	File-Sharing, Diskussionen mit Dozenten und Gruppenräumen
Remembering	Selbsttest	Fragebogen

tierte Programmierung; ca. 160 Stunden mit ca. 5–10 regulären Studierenden) gehalten. Zur Strukturierung unserer Lehre, verwendeten wir die Lernpyramide (siehe Abb. 3) und das Lernmodell von Race (siehe Abb. 4). Die Lernpyramide stellt ein Effektivitätsmaß je Lehr- und Lernmethode im Rahmen der Vorlesung dar. Die Lernkreise von Race stellen die Lernmethoden dar, welche für effektives Lernen durchlaufen werden.

Aktivierung

Wir stellten fest, dass Aktivierung und eine aktive Beteiligung der Studierenden sehr nützlich waren, um unsere Lehr- und Lernziele zu erreichen und Missverständnisse beim Verständnis der OOP-Terminologie und -Konzepte zu vermeiden [11]. Wir verwendeten verschiedene Methoden, um unsere Studierenden zu aktivieren und einzubeziehen. Wir aktivierten die Studierenden durch direkte Fragen und Diskussionen, Fragebögen (siehe Abb. 2) mit einer Frage

und Mehrfachantworten. Die Online-Fragen und -Diskussionen waren in der Online-Umgebung sehr gut möglich. Die Studierenden nahmen sehr oft am Online-Fragebogen teil. Sie beteiligten sich auch sehr aktiv an der Diskussion über ihre Fehler.

Wir nutzten auch die Bildschirmfreigabe- oder Remote-Funktion zur Aktivierung der Studierenden während der Online-Vorlesung (siehe Abb. 1). Die Studierenden konnten in die Rolle eines Dozenten schlüpfen, damit sie anderen Studierenden zeigen konnten, wie sie ihr Programm programmieren (die Lehrmethode heißt umgekehrter Klassenraum [12, 20, 21]). Der Dozent konnte einige Änderungen am Code vornehmen oder den Code mit den Studenten debuggen.

Online-Coding

Die Online-Coding war die effektivste Lehrmethode im Hinblick auf die Evaluations- und Feedback-Phase des Kur-

Abb. 2 Definitionen des Fragebogens und Antworten des Fragebogens mit Adobe Connect

The image shows four panels from an online survey tool. The top-left panel, titled 'Definition der Fragen', shows a question about the meaning of console, tracing, and logging. The top-right panel, titled 'Onlineumfrage', shows the same question with three multiple-choice options and a progress bar. The bottom-left panel, titled 'Antworten', shows a list of four student answers. The bottom-right panel, titled 'Ergebnisse', shows the same question with a progress bar and a list of three options, with the first option selected.

ses am Ende des Semesters. Wir denken auch, dass die virtuelle Lehr- und Lernumgebung diese Methode ideal unterstützt, da der Dozent und die Studierenden den Lernkontext [2] nicht gewechselt haben. Die Studierenden verwenden in ihrem Betriebssystem (OS) die Lehrsoftware Adobe Connect und die Eclipse Integrated Development Environment (IDE)³ für Entwickler im gleichen Lehr- und Lernkontext [2].

Wir haben Online-Coding in jedem OOP-Thema in der Vorlesung verwendet und unsere Code-Beispiele über den Debugger der Eclipse-IDE gedebuggt (siehe Abb. 5). Wir haben den Debugger verwendet, um zu zeigen, wie Zeichenketten gespeichert werden, z.B. an welcher Position ein Zeichen gespeichert wird. Wir haben den Debugger auch verwendet, um Zustände eines Objekts oder im Programm an einem bestimmten Zustand zu zeigen, sowie z.B. zu zeigen, wie sich der Datenstruktur-Stack in der Programmumgebung verhält.

Die Studenten sagten, dass das Online-Coding sehr hilfreich war, wenn das Online-Coding direkt nach der Vermittlung eines Programmierkonzepts durchgeführt wurde. Den Studenten einfach zu erklären, wie API-Elemente zu verwenden sind, ohne dies in einer IDE zu zeigen, war sehr unbefriedigend und frustrierend für sie, da sie die Wirkung und das Ergebnis z.B. einer Methode oder eines Programmierkonzepts nicht verstehen konnten.

³ <https://www.eclipse.org/downloads/packages/release/2019-09/r/eclipse-ide-java-developers>.

Gruppenräume

Wir bildeten Gruppen über die Gruppenfunktion von Adobe Connect, bei der wir die Schüler nach dem Zufallsprinzip in Gruppen mit einer Größe von 4–5 Mitgliedern unterteilten (siehe Abb. 1). Ihnen standen Pods als White-Board und andere zur Verfügung, um ein Programmierproblem zu lösen und ein Programmierkonzept, z.B. eine Klasse, zu visualisieren. Sie wurden mit Aufgaben wie dem Verständnis der Verwendung der API-Elemente oder eines kleinen Programms mit mehreren Klassen und Methoden konfrontiert und mussten die Lösung gemeinsam erarbeiten. Nach einem Gruppentreffen präsentierte jede Gruppe ihre Ergebnisse den anderen Gruppenmitgliedern. Wir fanden, dass diese Methode sehr nützlich war, um ein tieferes Verständnis der Programmierkonzepte zu erhalten.

Fragebögen

Eine Vorlesung dauerte 3 Stunden und 15 Minuten. In der Hälfte der Vorlesung (nach ca. zwei Stunden) machten wir eine Pause von 10 Minuten. Nach der Pause wurden den Studierenden mehrere Fragen zu den Vorlesungsinhalten zu den zwei Stunden der Vorlesung gestellt. Wir werteten die Antworten direkt aus und gaben ein positives Feedback (mit Lob und Anerkennung [22]). Wir konnten so die missverstandenen Konzepte gleich nach ihrer Erläuterung klären. Wir konnten auch verstehen, welche Konzepte der OOP in Java nicht verstanden wurden und intensiver behandelt werden mussten.

Chat für Fragen und Weblinks

Wir nutzten die Chat-Funktion von Adobe Connect während der Vorlesung als Gelegenheit, zusätzliche Links oder Informationen auszutauschen, die die Vorlesung erweitern. Während der Online-Codingsitzungen hatten wir auch API-Methoden verwendet. Wir haben Sie mit den Studierenden gemeinsam nachgeschlagen, indem wir die Weblink-Funktion von Adobe Connect nutzten. Bei der Verwendung der Web-Link-Funktion von Adobe Connect wurden die Studierenden von ihrem Dozenten und Adobe Connect in ihrem Webbrowser zur entsprechenden Java-API-Website geleitet. So konnten die Studierenden lernen, wie man eine Online-Ressource für Entwicklungsprobleme zu dem Zeitpunkt nutzen kann, zu dem sie im Entwicklungskontext [23, 24] benötigt wird. Diese Episode [25] und wir als Modell [26] konnten dadurch eine Problemlösung mittels externer Ressource aufzeigen, was den Lerneffekt verstärkte und die Handlungsfähigkeit in solchen Problemkontexten förderte.

Zusammenfassung der Ergebnisse

Unsere Ergebnisse zur Wirksamkeit unseres Unterrichts orientierten sich an der Lernpyramide (siehe Abb. 3) und dem Modell von Race (siehe Abb. 4) und zeigten deren Effektivität auf. Wir versuchten wie in der Lernpyramide aufzeigt, die Studierende direkt am Lernprozess teilzunehmen. Wie lasen die Studierende miteinander diskutieren und Sie konnten auch andere in der gleichen Gruppe mit Begleitung des Dozenten unterrichten. Insbesondere das Üben durch tun in diesem problemorientierten Fach, motivierte die Studierende, trotz der Komplexität des Faches OOP weiter die Inhalte aktiv zu studieren. Wie nach Race (siehe Abb. 4)

Abb. 3 Lernpyramide nach National Training Laboratories (NTL), Bethel Maine [7, 8]

Abb. 4 Lernprozess nach Phil Race [10, 19]

wollten die Studierenden lernen, in OOP zu programmieren. Wir erklärten ihnen OOP-Konzepte und -Methoden, und lieferten ein OOP-Beispiel, das die Wirkung der angewandten Methoden und OOP-Konzepte aufzeigte. Wir gaben nach den Fragebögen Feedback, und die Studierenden konnten ihre Problemlösungen in den Gruppenräumen und in der Online-Diskussion mit dem Dozenten verbalisieren.

Threats to Validity und Einschränkungen

Die Idee der Vorlesung besteht darin, ein Studienbuch für das Selbststudium zu erweitern. Bei anderen Vorlesungen, in denen OOP gelehrt wird und kein Studienbuch vorhanden ist, könnten unsere erklärten Methoden und Erfahrungen weniger Gültigkeit haben, da die Studierende vermittelte Inhalte nicht nochmals nachlesen können. Basierend auf den Ergebnissen der Lernpyramide [7] erwarten wir ähnliche Ergebnisse. Alle Studenten hatten einen starken technischen beruflichen Hintergrund, was auch die Ergebnisse in anderen OOP-Vorlesungen mit Studenten, die einen anderen Hintergrund haben, beeinflussen könnte.

Die gesamte Modulvorlesung zu OOP hat eine Länge von insgesamt 40 Stunden. Wir konnten im Rahmen dieser Zeit nur begrenzt auf alle Übungen im Studienheft Bezug nehmen und ein Feedback geben. Wir boten den Studierenden an, offenstehende Übungen und aufgetretene Probleme, die im Selbststudium des Übungsheftes aufkamen, in der Vorlesung zu besprechen. Wir fanden es sehr nützlich, diese Einschränkung mit Lernmöglichkeiten von On-

Abb. 5 Debugging in the Eclipse IDE with Adobe Connect (sharing screen pod)

line-Lernplattformen wie code.org oder codecademy.com zu unseren Vorlesungsinhalten zu ergänzen. Die Lernplattformen ermöglichen den Studierenden direktes Feedback zu den behandelten Themen und den Vorlesungsinhalten. Sie konnte zudem zu vielen Umsetzungsproblemen [28] in OOP weitere Vorschläge erhalten.

Wir empfehlen den Studierenden auch, interne Foren mit ihren Kommilitonen zu nutzen, die nicht regelmäßig genutzt wurden. Wir stellten fest, dass die Studierenden es sich mehr wünschen, sich innerhalb der Vorlesung mit dem Dozenten (wie auch Hölbl et al. für eine MOODLE-Vorlesung [5] herausfanden) oder in den Gruppenräumen auszutauschen.

Schlussfolgerung

Wir fanden heraus, dass im Fernlehren und Fernlernen in OOP Inklusion und Aktivierung die Schlüsselmethoden sind. Wir aktivierten die Studenten, indem wir Gruppenräume einrichteten und Diskussionen mit den Studenten führten oder Fragebögen zur Verfügung stellten. Dadurch blieben die Studenten während der gesamten Vorlesung motiviert. Wir zeigten in einer IDE die gelehrt Programmiermethoden und -konzepte bröckchenweise und lieferten zu abstrakten Erklärungen konkrete OOP-Beispiele. Dies war sehr nützlich, um Frustrationen bei den Studierenden beim Lernen von OOP zu verringern und die Zufriedenheit unserer Studenten zu erhöhen. OOP-Vorlesungen, die in einer virtuellen Lern- und Lernumgebung abgehalten werden, könnten die hohe Abbruchrate von OOP-Kursen mit der Anwendung einer Vielzahl von Lehr- und Lernmethoden im gleichen Lernkontext verringern, z. B. durch den Einsatz von Adobe Connect mit seinen Pods und Funktionalitäten.

Danksagung

Wir danken der DIPLOMA Fachhochschule Bad Sooden-Allendorf und Prof. Dr. Michael Namokel für die Veröffentlichung dieser Forschungsergebnisse. Wir danken auch Haitam Ben Yahia und Felix Schmidt für das Korrekturlesen der Arbeit.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access Dieser Artikel wird unter der Creative Commons Namensnennung 4.0 International Lizenz veröffentlicht, welche die Nutzung, Vervielfältigung, Bearbeitung, Verbreitung und Wiedergabe in jeglichem Medium und Format erlaubt, sofern Sie den/die ursprünglichen Autor(en) und die Quelle ordnungsgemäß nennen, einen Link zur Creative Commons Lizenz beifügen und angeben, ob Änderungen vorgenommen wurden.

Die in diesem Artikel enthaltenen Bilder und sonstiges Drittmaterial unterliegen ebenfalls der genannten Creative Commons Lizenz, sofern sich aus der Abbildungslegende nichts anderes ergibt. Sofern das betreffende Material nicht unter der genannten Creative Commons Lizenz steht und die betreffende Handlung nicht nach gesetzlichen Vorschriften erlaubt ist, ist für die oben aufgeführten Weiterverwendungen des Materials die Einwilligung des jeweiligen Rechteinhabers einzuholen.

Weitere Details zur Lizenz entnehmen Sie bitte der Lizenzinformation auf <http://creativecommons.org/licenses/by/4.0/deed.de>.

Literatur

1. Kim K-J, Bonk CJ (2006) The future of online teaching and learning in higher education. Educ Q 29(4):22–30
2. Anderson T (2004) Teaching in an online learning context. Theory Pract Online Learn 273:276–279. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.131.9849&rep=rep1&type=pdf>
3. Yuan L, Powell S (2013) Moocs and open education: implications for higher education
4. Giraffa LM, Moraes MC, Uden L (2014) Teaching object-oriented programming in first-year undergraduate courses supported by virtual classrooms. In: The 2nd international workshop on learning technology for education in cloud. Springer, Berlin Heidelberg, S 15–26

5. Hölbl M, Welzer T (2010) Students' feedback and communication habits using moodle. *Elektronika Elektrotechnika* 102(6):63–66
6. Khalil H, Ebner M (2014) Moocs completion rates and possible methods to improve retention-a literature review, in *EdMedia+ innovate learning*. Association for the Advancement of Computing in Education (AACE), S 1305–1313. Konferenz: World Conference on Educational Multimedia, Hypermedia and Telecommunications, Land: Finnland; <https://graz.pure.elsevier.com/de/publications/moocs-completion-rates-and-possible-methods-to-improve-retention>
7. Lalley J, Miller R (2007) The learning pyramid: does it point teachers in the right direction. *Education* 128(1):16
8. Kumar A (2009) Personal, academic and career development in higher education: SOARing to success. Routledge, London. <https://doi.org/10.4324/9780203938348>
9. Race P (2019) The lecturer's toolkit: a practical guide to assessment, learning and teaching. Routledge, London. ISBN 978-0367182267
10. Race P (2014) Making learning happen: A guide for post-compulsory education. SAGA Publications. ISBN 978-1446285961
11. Holland S, Griffiths R, Woodman M (1997) Avoiding object misconceptions. In: *Proceedings of the twenty-eighth SIGCSE technical symposium on Computer science education*, S 131–134
12. Ramsden P (2003) Learning to teach in higher education. Routledge, London. <https://doi.org/10.4324/9780203507711>
13. Godden DR, Baddeley AD (1975) Context-dependent memory in two natural environments: on land and underwater. *Br J Psychol* 66(3):325–331
14. Jung M (2015) Fernunterrichtsstistik 2014 veröffentlicht. <https://www.fernstudium-infos.de/fernstudium-rundschau/fernunterrichtsstatik-2014-ver%C3%B6ffentlicht-r52/>. Zugegriffen: 14.05.2020
15. DK, Hemmings J (2018) How psychology works: applied psychology visually explained
16. Forehand M (2010) Bloom's taxonomy. *Emerg Perspect Learn Teach Technol* 41(4):47–56
17. (2019) U. Academic Computing Services Columbia University, What are pods? <https://sites.google.com/a/tc.columbia.edu/adobe-connect-help/what-are-pods>. Zugegriffen: 14.05.2020
18. Adobe Inc (2019) Start, attend, and manage adobe connect meetings and sessions. <https://helpx.adobe.com/adobe-connect/using/starting-attending-meetings.html>. Zugegriffen: 14.05.2020
19. Race P (2019) Updated powerpoint of 'ripples model'. <https://phil-race.co.uk/2016/04/updated-powerpoint-ripples-model/>. Zugegriffen: 14.05.2020
20. Gannod G, Burge J, Helmick M (2008) Using the inverted classroom to teach software engineering. In: *2008 ACM/IEEE 30th International Conference on Software Engineering*. IEEE, New York, S 777–786. <https://dl.acm.org/doi/10.1145/1368088.1368198>
21. Talbert R (2012) Inverted classroom. *Colleagues* 9(1):7
22. Waldersee R, Luthans F (1994) The impact of positive and corrective feedback on customer service performance. *J Organiz Behav* 15(1):83–95
23. Maalej W, Tiarks R, Roehm T, Koschke R (2014) On the comprehension of program comprehension. *ACM Trans Softw Eng Methodol* 23(4):1–37
24. Maalej W, Ellmann M, Robbes R (2017) Using contexts similarity to predict relationships between tasks. *J Syst Softw* 128:267–284
25. Tulving E (1985) How many memory systems are there? *Am Psychol* 40(4):385
26. Bandura A (2008) Observational learning, *The international encyclopedia of communication*
27. Cowan N (2001) The magical number 4 in short-term memory: a reconsideration of mental storage capacity. *Behav Brain Sci* 24(1):87–114
28. Heeb H (2001) Overcoming the problem of cognitive load in object-oriented programming by microworlds. <http://heeb.ch/java/microworlds/oopmmal.pdf>