Visualisierung einfacher Algorithmen für den Informatikunterricht

Unterprogrammtechnik

Wissenschaftliche Arbeit
im Fach Informatik

Lehramt an Gymnasien

eingereicht von
Lehmann, Sebastian
geboren am 03.03.1985

Technische Universität Dresden
Fakultät Informatik
Institut für Software- und Multimediatechnik
Fachgebiet Didaktik der Informatik/Lehrerbildung

Gutachter:
Dr. Holger Rohland
Klaus Thuß

Dresden, September 2009

Technische Universität Dresden

Fakultät Informatik

Aufgabenstellung für eine wissenschaftliche Arbeit

Name, Vorname des Studenten: Lehmann, Sebastian
Immatrikulationsnummer: 3128000

Thema: "Visualisierung einfacher Algorithmen für den Informatikunterricht"

Zielstellung:

Einfache Algorithmen bilden die Grundlage für das Programmieren. Obwohl die meisten Funktionen und Abfolgen logisch und trivial sind, bereiten sie den meisten Anfängern Schwierigkeiten. Um diese zu beheben ist im Rahmen einer Diplomarbeit ein Werkzeug zur Visualisierung solcher grundlegenden Programmstrukturen geschaffen worden.

In der hier anzufertigenden Arbeit soll das Werkzeug weiter entwickelt werden. Dazu ist das erstellte Werkzeug zunächst mit einer größeren Gruppe von Lehrern und Schülern zu evaluieren. Unter Berücksichtigung der Evaluierungsergebnisse sind dann weitere Module zu entwickeln, wobei insbesondere die Visualisierung von Unterprogrammtechniken (Funktionen und Prozeduren) und deren Spezifik (Wert- und Variablenparameter) veranschaulicht werden sollen. Abschließend sollen auch die neu entwickelten Module einem Praxistest unterzogen werden.

Im Einzelnen werden folgende Ergebnisse angestrebt:

- Erstellen einer vergleichenden Übersicht mit anderen existierenden Lösungen für die Visualisierung einfacher Algorithmen
- Erweiterung der bestehenden Lösung durch Visualisierungen zum Themenkomplex "Unterprogrammtechniken" unter Einsatz des Werkzeugs Flash
- Evaluierung der neu erstellten Module
- Integration in bereits bestehende Website für Lehr- und Lernmaterial auf dem Sächsischen Bildungsserver

Betreuer:	Dr. rer. nat. Holger Rohland
Betreuender Hochschullehrer:	Prof. Dr. paed. habil. Steffen Friedrich
Institut:	Software- und Multimediatechnik
Beginn am:	01.07.2009
Einzureichen am:	01.10.2009

Inhaltsverzeichnis

1 Einführung

Einen Satz zu sprechen, ein Glas Wasser zu trinken, oder ein Foto mit dem Fotoapparat zu schießen, jede dieser unterschiedlichen Tätigkeiten ist für uns eine Selbstverständlichkeit. Dabei besitzen wir diese Fertigkeiten nicht von Geburt an, sondern lernen den Umgang mit diesen Dingen im Laufe unserer persönlichen Entwicklung. Manchmal treffen wir jedoch auch auf Probleme, die sich von uns nur schwer oder überhaupt nicht lösen lassen. Da der Mensch aber erfinderisch ist, versucht er sich Hilfsmittel zu erschaffen, die schwere oder scheinbar unlösbare Probleme lösen können. Durch die Erfindung von Computersystemen lassen sich vielfältige Probleme des Alltags, aber auch spezieller, wissenschaftlicher Art lösen. Dazu werden Beschreibungen benötigt, die das Problem so genau wie möglich erläutern und die von dem verwendeten System verstanden und abgearbeitet werden können. Die Beschreibung wird dann in einer sogenannten Programmiersprache verfasst, die der Computer in seine Sprache übersetzen und auswerten kann.

Eine Programmiersprache ist, im Gegensatz zu unserer Umgangssprache, in gewisser Form eine abstrakte Sprache, die wir zur gegenseitigen Kommunikation nicht gebrauchen würden, obwohl sie sogar genauer Ausdrücken könnte, was wir sagen wollen. Um eine Programmiersprache beherrschen zu können, sind abstrakte Denkweisen notwendig. Abstrakte Denkweisen sind meist nicht nur für Schüler schwer nachzuvollziehen.

Die oben genannten Vorgänge, einen Satz zu sprechen, ein Glas Wasser zu trinken, oder ein Foto zu schießen, werden bei jeder Durchführung dieser Tätigkeiten nicht neu von uns erlernt, sondern sind bereits in unserem Bewusstsein verankert und werden zu gegebener Zeit abgerufen. Denn es wäre ein erheblicher Zeitaufwand, das Trinken eines Glases Wasser jedes Mal aufs Neue erlernen zu müssen. Ein ähnliches Konzept gibt es auch in der Programmierung. Die sogenannte Unterprogrammtechnik ermöglicht, dass oft genutzte Anweisungen, die einmal programmiert wurden, schnell abrufbar gemacht werden.

Unterprogrammtechnik ist in der Programmierung ein Konzept zur Strukturierung des Programmcodes. Diese Technik findet sowohl hardwarenah auf Maschinenebene Anwendung, als auch in fast allen gängigen Programmiersprachen. Man unterscheidet zwischen sogenannten Funktionen und Prozeduren. Der Sinn der Unterprogrammtechnik ist es, an geeigneter Stelle des Hauptalgorithmus auf häufig verwendete oder bereits bekannte Programmkonstrukte zuzugreifen.

Unterprogramme können als eine Art Bibliothek angesehen werden. Will man ein Problem lösen, sucht man sich zunächst aus der Bibliothek bereits vorhandene Bücher zu einem Schwerpunkt, die zum Lösen des Problems beitragen können. Es reicht dann vollkommen aus, einen Verweis auf das entsprechende Buch an geeigneter Stelle in der Lösungsbeschreibung anzugeben. Somit können die Inhalte der Bücher bzw. Unterprogramme mehrfach verwendet werden und müssen nicht neu geschrieben werden. Diese Idee ist an sich leicht zu verstehen, die Umsetzung in einer Programmiersprache ist jedoch auf dem ersten Blick nicht leicht zu verstehen. Denn man wird im Laufe seines Lebens nicht immer das gleiche Wasserglas mit der gleichen Menge Wasser füllen, oder den gleichen Satz sprechen. Selbst der Fotoapparat wird einmal ein anderer sein. Deshalb ist es möglich, bekannte Vorgänge auf unterschiedliche Ausgangsbedingungen anzupassen. Dies ist ebenfalls in der Unterprogrammtechnik möglich, durch Übergabe von Parametern an die Vorgangsbeschreibung.

Neulingen auf dem Gebiet der Programmierung fällt es nicht leicht, sich in die abstrakten Beschreibungen der Programmierung hineinzuversetzen, da die abstrakte Denkweise erst noch entwickelt werden muss.

Aus diesen Gründen soll eine verständliche Visualisierung der Unterprogrammtechnik das Kernthema der Betrachtungen der zugrundeliegenden wissenschaftlichen Arbeit sein. In diesem Rahmen ist bereits in einer Diplomarbeit ein Werkzeug erstellt worden, das die Einführung der Unterprogrammtechnik dem Lernenden, speziell Schüler, erleichtern soll. Dieses Werkzeug wird durch diese Arbeit weiter ausgebaut und verbessert. Die Visualisierung von grundlegenden Programmstrukturen ist bereits im Werkzeug umgesetzt worden.

1.1 Ziel und Aufbau der Arbeit

Das Ziel dieser wissenschaftlichen Arbeit ist die Erstellung und Einbindung geeigneter Module in das bereits vorhandene Werkzeug „Programmieren .. Begreifen“, um eine möglichst einfache und verständliche Einführung des Themas Unterprogrammtechnik für Schüler und Lehrer bereitzustellen. Das Werkzeug soll sowohl im Selbstlernprozess unter tutorieller Betreuung, als auch unter Anleitung einer Lehrperson, zum Beispiel über einen Projektor, einsetzbar sein.

Um dieses Ziel zu erreichen, sollen dazu in den folgenden Kapiteln zunächst die folgenden Fragen geklärt werden:

- *Warum ist Programmierung im Schulunterricht wichtig?*
- *Welche Anforderungen werden im Informatikunterricht gestellt?*
- *Welche Probleme treten bei der Einführung in die Programmierung auf?*
- *Wie könnte ein effektiver Einstieg aussehen?*
- *Was ist Unterprogrammtechnik und wie funktioniert sie?*
- *Welchen Nutzen hat die Unterprogrammtechnik?*
- *Welche Werkzeuge gibt es bereits und wie ist die Qualität dieser einzuschätzen?*

Dazu wird in Kapitel 2 geprüft, wie Programmierung, speziell Unterprogrammtechnik, in der Schule bzw. im Informatikunterricht integriert ist, und welche Forderungen an die Schüler, aber auch an die Lehrer, gestellt werden. Speziell wird dabei auf die Bildungsstandards des Faches Informatik sowie auf den Lehrplan des Freistaates Sachsen Bezug genommen. Zudem soll ein möglichst guter Einstieg in die Programmierung aufgezeigt werden, unter Berücksichtigung häufig auftretender Tücken und Fehlerquellen.

Der Aufbau und die Funktionsweise von Unterprogrammtechnik bilden in Kapitel 3 einen weiteren Schwerpunkt der Arbeit.

Der Nutzen von Lernprogrammen im Informatikunterricht soll im weiteren Verlauf betrachtet werden, um den Sinn des Einsatzes von Lernsoftware im Unterricht zu überprüfen. Um die Qualität des Werkzeugs zu sichern, werden verschiedene existierende Lösungen in Kapitel 2.3 betrachtet und deren Stärken sowie Schwächen

herausgearbeitet und miteinander verglichen. Der Vergleich soll auch Aufschluss darüber geben, warum die Erstellung des Werkzeugs von Bedeutung ist. Dabei werden die Programme auf ihre Gebrauchstauglichkeit für den Informatikunterricht hin untersucht. Aber sie werden auch nach der Implementierung und Umsetzung der Unterprogrammtechnik untersucht und beurteilt.

Nach beantworten der oben genannten Ziele, folgt die Vorstellung des Werkzeugs. Neben der Implementierung weiterer Module für die Unterprogrammtechnik wurde anhand der Evaluierungsergebnisse des ursprünglichen Werkzeugs eine effektivere sowie einfachere Bedienung geschaffen, um den Lernprozess weiter zu erleichtern und die Qualität des Werkzeugs zu steigern. Der Aufbau und die Funktionsweise werden im Kapitel 4 erklärt, sowie die Neuerungen, ausgehend von vorhandenen Evaluierungsergebnissen, präsentiert. Zur Qualitätssicherung wird das neue Werkzeug ebenfalls einer Evaluierung unterzogen.

Das Kapitel 5 fasst die Ergebnisse der Arbeit noch einmal kurz zusammen. Abschließend wird in Kapitel 6 ein Ausblick auf zukünftige Erweiterungen des Werkzeugs, auch mit Blick auf andere Lernprogramme, gegeben.

Das hiermit entstandenen, weiterentwickelte Werkzeug wird Lehrern und Schülern auf der Webseite des Sächsischen Bildungsservers unter der Rubrik Lehr- und Lernprogramme frei zugänglich bereitgestellt.

5 Zusammenfassung

Ziel der wissenschaftlichen Arbeit war das Erstellen weiterer Module für die Visualisierung von Unterprogrammtechnik zum unterstützenden Einsatz im Informatikunterricht. Um dieses Ziel zu erreichen, wurden zunächst verschiedene Aspekte analysiert. So wurde aufgezeigt, dass die Programmierung quasi eine Notwendigkeit für das Verständnis informatischer Grundbildung darstellt, da sie ein wichtiger Teil eines Problemlöseprozesses ist, der die gesamten Prozess- und Inhaltsbereiche des Faches abdeckt. Ebenfalls konnte gezeigt werden, dass die Programmierung im Zusammenhang mit Problemlöseprozessen mindestens genauso positive Effekte auf die Entwicklung der Schüler erzielt, wie beispielsweise Problemlöseprozesse im Mathematikunterricht. Die Auseinandersetzung mit Problemlöseprozessen erfordert und fördert einen Umgang des präzisen Ausdrucks und Denkens der Schüler, was im späteren Leben sehr hilfreich ist.

Empfehlungen zu den Anforderungen des Informatikunterrichts sind in den Bildungsstandards der Gesellschaft für Informatik e. V. festgelegt, um qualitativ hochwertigen Unterricht deutschlandweit zu sichern. Der Einsatz von unterrichtsbegleitenden Werkzeugen kann in diesem Sinne die Qualität des Unterrichts verbessern. In Sachsen ziehen sich Problemlöseprozesse durch alle Jahrgangsstufen des Informatikunterrichts. Im Interesse eines begründeten Anspruchs der Informatik als Schulfach muss die weit verbreitete Vorstellung, Informatik mit dem stupiden Schreiben von Programmen gleichzusetzen, aus den Köpfen der Gesellschaft verdrängt werden.

Weiterhin ist es wichtig für den Lehrenden, sich mit möglichen Problemen bei der Einführung in die Programmierung auseinanderzusetzen. Denn es können vermehrt Probleme beim Umgang mit (vor allem professionellen) Programmierumgebungen auftreten, die dem Lernprozess der Schüler entgegenwirken. Eine Zuhilfenahme von Lernprogrammen und unterstützenden Werkzeugen kann in Hinblick auf eine höhere

Motivation und ein besseres Verständnis der Schüler sinnvoll sein, da dadurch eine Reduzierung primärer und sekundärer Problemfelder erreicht werden kann.

Ebenfalls konnte geklärt werden, dass unter dem Begriff der Unterprogrammtechnik ein Konzept der strukturierten Programmierung zu verstehen ist, deren Vorteile in der besseren Wartung, Effizienz und Weiterentwicklung eines Projektes liegen. Die generelle Unterteilung von Unterprogrammen in Prozedur und Funktion ist fundamental für den Informatikunterricht. Ebenfalls die Funktionsweise von verschiedenen Typen von Übergabeparametern ist eine wichtige Grundlage dieser Technik und spielt im Unterricht eine wichtige Rolle. Dazu erfolgt zumeist eine Beschränkung auf die Typen Werteparameter und Variablenparameter. Diese Funktionalitäten der Unterprogrammtechnik und Unterschiede in ihren Details zu veranschaulichen ist eine schwierige Aufgabe. Aufgrund der Vielzahl an Begriffen zur Unterscheidung von Unterprogrammtypen, Parameterarten bzw. Übertragungsformen ist es für Schüler trotz der Reduzierung schwierig, den Überblick zu behalten.

Werkzeuge, die eine Unterstützung zu diesem Teilgebiet leisten, konnten nicht ausfindig gemacht werden. Dafür gibt es eine Anzahl an guten Werkzeugen, die bekannte Sortierverfahren veranschaulichen. Die Implementierung der Unterprogrammtechnik im Werkzeug „Programmieren .. Begreifen“ ist somit ein Novum. Das erweiterte und überarbeitete Werkzeug dieser Arbeit kann dem Lehrer als auch dem Schüler an dieser Stelle die Einführung in die Programmierung erleichtern, da es distanziert und unabhängig von gängigen Programmiersprachen und Lernprogrammen arbeitet.

Zur Qualitätssicherung wurde ein Evaluationsbogen erstellt, der an Lehrer, Referendare und Studenten der Informatik verteilt wurde. Eine positive Beurteilung ist zum einen anhand der durchschnittlichen Bewertung mit der Schulnote 1,77 ersichtlich. Des Weiteren konnte die Benutzerfreundlichkeit weiter verbessert werden und auch die neuen Module zur Unterprogrammtechnik konnten fast alle Evaluationsteilnehmer positiv überzeugen, dass Werkzeug (vielleicht) im Unterricht einzusetzen.

6 Ausblick

Lernprogramme und tutorielle Werkzeuge werden in der Zukunft immer mehr an Bedeutung gewinnen, zum einen durch höhere Verfügbarkeit über das Internet, als auch durch höhere Nachfrage. Dies ist natürlich der schnellen Entwicklung von geeigneten Werkzeugen zu verdanken. Lernprogramme und Werkzeuge werden aber nicht nur für den Informatikunterricht von Vorteil sein. Beispielsweise steht für das Fach Mathematik bereits eine Menge an Werkzeugen frei zur Verfügung. Jedoch fehlen zumeist noch visuelle Unterstützungen. Sicherlich wird es immer Sachverhalte geben, die nicht einhundertprozentig visuell dargestellt werden können, aber der aktuelle Stand der Technik erlaubt uns nun Fortschritte in dieser Richtung zu erzielen. Das hier entwickelte Werkzeug kann als einer der Vorreiter in Sachen Visualisierung von unterrichtstheoretischen Inhalten angesehen werden, vor allem für die verständliche visuelle Einführung von einfachen Strukturen von Algorithmen. Das Werkzeug besitzt noch viel Potential zur Weiterentwicklung, um einen noch besseren und vielfältigeren Einsatz zu ermöglichen. Aufgrund der internen Programmstrukturen ist es relativ einfach möglich mit vorhandenen Konstrukten, wie zum Beispiel Schleifen oder Unterprogrammen, weitere Module zu entwickeln. Das schwierigste dabei ist lediglich die Implementierung neuer Animationen, die parallel zum Quelltext ablaufen. Die Implementierung weiterer Konstrukte ist ebenfalls möglich. Beispielsweise können die vorhandenen Konstrukte der Unterprogrammtechnik genutzt werden, um die Rekursion in neuen Modulen zu visualisieren. Rekursion findet häufig Anwendung als Strategie zum Problemlösen und wird ebenfalls im Schulunterricht thematisiert. Ein Editor für den Anwender, mit dem er vorhandene Strukturen zu einem Programm zusammensetzt, wäre aufgrund der Programmstruktur ebenfalls denkbar. Denn um einen Quellcode für ein Modul zu erstellen, reicht ein Verweis mit gewünschten Parametern auf bereits vorhandene Strukturen aus. Des Weiteren könnte auch die Implementierung eines visualisierten Struktogramms zu jedem Modul sinnvoll sein, welches neben der Animation und dem Belegungsprotokoll zugeschaltet werden kann. Denkbar wäre auch eine Vergleichsmöglichkeit zweier Module, um die Gemeinsamkeiten und Unterschiede noch deutlicher hervorheben zu können.

B Abbildungsverzeichnis

C Tabellenverzeichnis

D Literaturverzeichnis

Appelrath, Hans-Jürgen. 2002. *Starthilfe Informatik.* 2. Auflage. s.l. : B.G. Teubner Verlag, 2002. ISBN 978-3-519-10241-0.

Barbu, Andreea, et al. 2009. Universität Oldenburg. [Online] Abgerufen am 28. August 2009. http://olli.informatik.uni-oldenburg.de/fpsort/.

Barth, Armin P. 2003. *Algorithmik für Einsteiger : für Studierende, Lehrer und Schüler in den Fächern Mathematik und Informatik.* Braunschweig : Vieweg & Teubner, 2003. ISBN 3528031964.

Barth, Peter, et al. 2000. *Kleiner Leitfaden Informatik und ihre Anwendungen.* [Hrsg.] Lutz Engelmann. Berlin : Paetec, Gesellschaft für Bildung und Technik mbH, 2000. S. 313. ISBN 3-89517-615-X.

Baumann, Rüdeger. 1992. *Informatik für der Sekundarstufe II - Band 1: Algorithmen, Datenstrukturen, Projekte.* Stuttgart : Ernst Klett Schulbuchverlag, 1992. Bd. 1. ISBN 3-12-717741-0.

Braun, Wolfgang. 1993. *Strukturiertes Programmieren mit TURBO-PASCAL.* [Hrsg.] Gebrüder Grimm. Darmstadt : Winklers Verlag, 1993. Bd. 1. ISBN 3-8045-4605-6.

Demel, Wolfgang. 2009. Österreichisches Schulportal. [Online] Abgerufen am 23. Juli 2009. http://lehrer.schule.at/demel/informatik/tp/upro_skriptum.pdf.

Deutsches Institut für Internationale Pädagogische Forschung. 2009. Deutscher Bildungsserver. [Online] Abgerufen am 28. August 2009. http://dbs.schule.de/db/fachlist.html?fach=4041&Suchwort=algorithmen.

Fechner, Siegfried, et al. 1997. *Lehr- und Übungsbuch Informatik.* [Hrsg.] Christian Horn und Immo O. Kerner. Dresden, Dublin : Fachbuchverlag Leipzig im Carl Hanser Verlag München Wien, 1997. Bd. 3. ISBN 3-446-18699-9.

Gesellschaft für Informatik (GI) e. V. 2008. Grundsätze und Standards für die Informatik in der Schule - Bildungsstandards Informatik für die Sekundarstufe I. *Beilage zu LOG IN.* 2008, 28. Jg. (2008), Heft Nr. 150/151.

Gorny, Peter und Faltin, Nils. 2009. Universität Oldenburg - Heapsort SALA. [Online] Abgerufen am 28. August 2009. http://olli.informatik.uni-oldenburg.de/heapsort_SALA/deutsch/inhalt.html.

Horn, Christian, Kerner, Immo O. und Forbrig, Peter, [Hrsg.]. 2003. *Lehr- und Übungsbuch Informatik.* 3., völlig neu bearbeitete Auflage. s.l. : Hanser Verlag, 2003. S. 404 ff. ISBN 3-446-22543-9.

Hubwieser, Peter. 2007. Didaktik der Informatik: Grundlagen, Konzepte und Beispiele. Berlin : Springer Verlag, 2007. ISBN 354072477X.

Humbert, Ludger. 2005. *Didaktik der Informatik mit praxiserprobtem Unterrichtsmaterial.* Wiesbanden : B.G. Teubner Verlag / GWV Fachverlage GmbH, 2005. ISBN 3-8351-0038-6.

Kohl, Lutz. 2009. ipuck.de. [Online] Abgerufen am 28. August 2009. http://ipuck.de/.

Kowalk, Wolfgang P. 2009. Universität Oldenburg - Animierte Algorithmen. [Online] Abgerufen am 29. August 2009. http://einstein.informatik.uni-oldenburg.de/20907.html.

Kron, Friedrich W. und Sofos, Alivisos. 2003. *Mediendidaktik.* München : UTB Reinhardt, 2003. S. 165 - 178. ISBN 3-8252-2404-X.

Levi, Paul und Rembold, Ulrich. 2002. *Einführung in die Informatik.* 4., aktualisierte Auflage. s.l. : Hanser Verlag, 2002. ISBN-10: 3-446-21932-3.

Meyers Lexikonredaktion. 1997. *Schüler Duden Informatik.* Mannheim : Dudenverlag, 1997. S. 384. ISBN 3-411-04483-7.

Ploedereder, Erhard. 2009. Universität Stuttgart - The Sort Algorithm Animator V1.0. [Online] Abgerufen am 28. August 2009. http://www.iste.uni-stuttgart.de/ps/Ploedereder/sorter/sortanimation2.html.

Rost, Detlef H., [Hrsg.]. 2001. *Handwörterbuch Pädagogische Psychologie.* 2. überarbeitete und erweiterte Auflage. Weinheim : Psychologische Verlagsunion, Verlagsgruppe Beltz, 2001. S. 555-560. ISBN 3-621-27491-X.

Sächsischen Staatsministeriums für Kultus und Technischen Universität Dresden. 2009. Sächsischer Bildungsserver. [Online] Abgerufen am 17. September 2009. http://www.sn.schule.de/.

Sächsisches Staatsministerium für Kultus. 2004/2007. *Lehrplan Gymnasium Informatik.* 2004/2007.

Schindler, Claudia. 2008. *Visualisierung einfacher Algorithmen für den Informatikunterricht.* Fachbereich Informatik, Technische Universität Dresden. 2008. Diplomarbeit.

Selig, Bettina, Kern, Vera und Walther, Tilman. 2009. tilman.de. [Online] Abgerufen am 11. August 2009. http://www.tilman.de/uni/ws03/alp/index.php.

Sperber, Michael. 2009. DeinProgramm - Programmieren für alle. [Online] Abgerufen am 02. August 2009. http://www.deinprogramm.de.