Hausarbeit zur Abschlussprüfung für erweiternde Studie für Lehrer im Fach Informatik

Funktionsweise und Struktur verschiedener Sortierverfahren

Ein Unterrichtsprojekt zum Kursthema Programmierung im Fach Datenverarbeitung der 12. Klasse des beruflichen Gymnasiums Wirtschaft

Eingereicht beim: **Amt für Lehrerausbildung**
Dezernat Weiterbildung
Rothwestener Str. 2-14
34233 Fuldatal

Vorgelegt von: **StR Dipl.-Hdl. Olaf Ruff**
Neuhausstr. 33
61440 Oberursel

Inhaltsverzeichnis

Seite

1. Einleitung 2

2. Fachwissenschaftliche Orientierung 3
2.1. Zum Begriff des Algorithmus 3
2.2. Die Bedeutung des Sortierens 4
2.3. Merkmale und Prinzipien von Sortierverfahren 5
2.4. Die Idee des Software Engineering 7

3. **Didaktisch-methodische Orientierung** 9
3.1. Handlungsorientierung und Projektmethode 9
3.2. Spiralprinzip und Lernschleife 10
3.3. Informatische Trias 12

4. Unterrichtspraktische Umsetzung 14
4.1. Lehrbedingungen 14
4.1.1. Rechtliche Einordnung und Vorgaben 14
4.1.2. Schulische Rahmenbedingungen 15
4.1.3. Die Schüler des Grundkurses Datenverarbeitung 16
4.2. Planung, Durchführung und Reflexion der Unterrichtsprozesse 17
4.2.1. Einordnung der Leitziele Selbständigkeit, Selbsttätigkeit, Selbstreflexion 17
4.2.2 Ablaufstruktur 18
4.2.3 Unterrichtsprozess I (Problemanalyse, Lösungsverfahren) 21
4.2.3.1. Lernziele 21
4.2.3.2. Planung, Durchführung und Reflexion 22
4.2.4 Unterrichtsprozess II (Entwurf, Implementation, Test) 26
4.2.4.1. Lernziele 26
4.2.4.2. Planung, Durchführung und Reflexion 28

5. Evaluation des Unterrichtsprojektes 33

Literaturverzeichnis

Anhang

1. Einleitung

Das Schreiben von Programmen, gleichgültig auf welche Programmiersprachen sich bezogen wird, stellt für viele Schüler an unserer Schule einen hohen Schwierigkeitsgrad dar. Ein Problem liegt darin, dass die Elemente einer Sprache, die Syntax und Semantik, keine kontinuierliche Wiederholung und Anwendung bei den Lernenden finden. Somit werden basale Strukturen schnell wieder vergessen werden. Der Grund liegt in der „informatischen Sozialisation“ der meisten Schüler, welche den PC ausschließlich als konsumtives Medium (zum Spielen, Video schauen, Chatten, etc.) ansehen und kein Interesse in der dahinterliegenden Funktionsweise der Maschine zeigen. Diese Haltung widerspricht dem Ideal des aufgeklärten Menschen. Weitere Hürden stellen das Erkennen eines Problems, dessen *algorithmische* Lösung und vor allem die Implementation in einer problemorientierten Sprache dar. Hierbei geht es um die Entwicklung einer oder mehrerer Ideen. Es sind Kreativität, Logik und Ausdauer gefragt, keine sehr beliebte Kombination von Attitüden in der heutigen Schülerwelt, für die spätere Berufs- und Arbeitswelt oder das Studium umso relevanter. Unter diesen Gesichtspunkten darf der Informatikunterricht nicht verkommen „als Fach für wenige Programmier-Freaks“[1], sondern muss die zentralen Prinzipien und Arbeitsweisen der Informatik vermitteln, die Anwendungsbezüge in der Realität sowie die gesellschaftliche Bedeutung herausstellen. Gerade die Informatik „ist eine Wissenschaft, deren Aufgabe vorrangig im Lösen von Problem und im Vermitteln der Fähigkeit zum Problemlösen besteht“[2].

Basierend auf der dargestellten Situation stellt sich für mich die Aufgabe, anhand eines Unterrichtsprojektes unter Beachtung des gesetzlichen Bildungsauftrages und der Lehrpläne zu überprüfen, ob ein adäquater Handlungsrahmen für Lehrer geschaffen werden kann, um die beschriebenen Probleme zu reduzieren bzw. zu lösen. Fachwissenschaftliche und didaktisch-methodische Überlegungen sind für mich erste Orientierungen, um den durch die Lehrpläne vorgelegten inhaltlichen Handlungsrahmen in Lernprozesse zu transformieren. Die eigentliche Herausforderung dieser Arbeit besteht in der unterrichtspraktischen Umsetzung in bezug auf die Problemstellung und den Anspruch an Informatikunterricht.

---

[1] Baumann, 1996, S. 109.
[2] Kerner, 1990, S. 118.

2. Fachwissenschaftliche Orientierung

2.1. Zum Begriff des Algorithmus

Die Wissenschaft Informatik ist in einer langen Geschichte aus der Mathematik (theoretische Basis) und der Technik (materiell-energetische Basis) gewachsen.
Der Begriff des Algorithmus stammt aus der Mathematik und geht vermutlich auf den arabischen Gelehrten **Al Chawarzmi** zurück.[3] Die Eigenart eines Algorithmus zu erkunden, weckte das Interesse ganzer Generationen von Mathematikern, welches ihren ersten Höhepunkt bei Leibniz (1646-1716) fand. Dieser vertrat die Auffassung, dass alle algebraischen Probleme auf einer Maschine realisiert werden können. Dieser Irrtum wurde erst 300 Jahre später durch die Arbeiten von Gödel, Turing und Church aufgedeckt und führte zu einem präzisierten Algorithmusbegriff.[4]
Unter dem Begriff Algorithmus verstand man ursprünglich ein systematisches Rechenverfahren (z.B. das Heron-Verfahren zur Berechnung der Quadratwurzel). Im Laufe der Zeit wurde die Bedeutung erweitert, indem viele Vorgehensweisen des täglichen Lebens und der Arbeitswelt (Bearbeitungsvorschriften, Bedienungsanleitungen, Kochrezepte etc.) einbezogen wurden. Hierbei tritt das Problem auf, dass die Sprache, in der ein Algorithmus formuliert ist, im Hinblick auf Präzision unterschiedlich ist. Für den Menschen stellt dies in der Regel kein Problem dar, bei Computern muss die Vorgehensweise eindeutig sein. [5]
Sucht man nach einer maschinenbezogenen Definition, so findet sich in der Literatur zwar keine einheitliche Aussage, aber eine Schnittmenge ist erkennbar. Folgende Definition spiegelt diese Schnittmenge wieder:

Ein Algorithmus ist ein System von eindeutigen Anweisungen, die eine Maschine in einer genau festgelegten Befehlsreihenfolge zu endlich vielen und ausführbaren Operationen veranlassen.

Hieraus ergeben sich formale Eigenschaften, die ein Algorithmus aufweist:

- Endliche Folge von Einzelschritten (Terminierung)
- Alle Einzelschritte sind präzise, eindeutig, elementar und ausführbar
- Abbildung von der Menge der zulässigen Eingabedaten in die Menge der Ausgabedaten (Vollständigkeit).

[3] Vgl. Diemer, 1992, S. 59.
[4] Vgl. Breier, o.J., S. 2
[5] Vgl. Hahn/Dzewas, 1991, S. 5ff.

- Bei jeder zulässigen Eingabe wird die Ausführung nach endlich vielen Einzelschritten beendet (Finitheit)
- Auf die selben Eingabedaten erfolgt stets das selbe Ergebnis (Determinismus). [6]

Offen bleibt die Frage nach der Wirksamkeit (Effizienz) eines Algorithmus. Für Kerner ist ein Algorithmus „effizienter als ein anderer, wenn das gleiche Resultat mit geringerem Aufwand erzielt“[7] wird. Hierbei muss allerdings beachtet werden, wie weit effiziente Algorithmen zugleich effektiv (wirklich ausführbar) sind.
Algorithmen bilden die (logische) Grundlage für ein Programm, welches ein konkretes Objekt, nämlich einen Text, darstellt und in unterschiedlichen Programmiersprachen geschrieben werden kann. Als Leitsatz kann festgehalten werden: Zuerst kommt der Algorithmus, danach das Programm. Aus dieser Betrachtungsweise heraus erwächst die Notwendigkeit, gerade während der Programmierarbeit mit Schülern, diesen die Ideen die hinter Algorithmen stehen, näher zu bringen und die beschränkte Sicht, dass nur eine Lösung (*der Algorithmus*) existiert, aufzubrechen.

2.2. Die Bedeutung des Sortierens

Das Sortieren ist eine universell praktizierte, grundlegende Handlungsweise, die bereits von kleinen Kindern gelernt wird, und die Grundlage des Lernens und Behaltens durch Strukturieren darstellt.[8] Auch Schülern ist das Sortierproblem unmittelbar geläufig, weil sie mit diesem direkt oder indirekt in Berührung kommen, z.B. Kartenspiele, Sporttabellen, Namenslisten. In der Informatik nimmt das Sortieren von Wort- bzw. Zahlenmengen einen wichtigen Stellenwert ein, da häufig Daten bzw. Informationen ungeordnet vorliegen. Ziel des Sortiervorgangs ist es, eine bestimmte Ordnung (Reihenfolge) innerhalb einer Menge zu erzeugen.[9] Die Anzahl an Beispielen für sortierte Mengen nimmt in unserer Gesellschaft ständig zu. Die bekanntesten sortierten Mengen sind:

- Telefonbücher
- Lexika
- Wörterbücher

[6] Vgl. Bleßmann u.a., 2000, S. 74.
[7] Kerner, 1990, S. 73.
[8] Vgl. Roth, 1965, S.278.
[9] Vgl. Wirth, 1983, S. 77.

Wie aus den Beispielen erkennbar, kann das Sortieren nach unterschiedlichen Ordnungsmerkmalen (numerisch, alphabetisch etc.) stattfinden. Die Notwendigkeit einer Ordnungsrelation, die mathematisch leicht beschrieben werden kann[10], bleibt stets bestehen. Diese Überlegung führt zu der Erkenntnis, dass Sortieren eine Aufgabe bleibt, die keine Lösungsformel, wohl aber ein Lösungsverfahren erfordert und somit einen dynamischen Prozess darstellt.

Der Verwendungszweck sortierter Mengen liegt in der Vereinfachung und Beschleunigung einer späteren Suche in der geordneten Menge. Das Sortieren von Mengen nimmt heute einen Großteil der Rechenzeit in Anspruch. Schätzungen gehen davon aus, dass über 25% aller Rechenzeit im kommerziellen Bereich hierfür aufgewendet wird.[11] Daher ist man bestrebt immer schnellere, kompaktere und somit bessere Sortierverfahren zu finden.

2.3. Merkmale und Prinzipien von Sortierverfahren

Es existieren zahlreiche Sortieralgorithmen, die meistens aus der Analyse manueller Sortiertechniken stammen[12] und auf wenige grundlegende Verfahren zurückzuführen sind:[13]

- Sortieren durch Austausch (Bubblesort, Quicksort)
- Sortieren durch Auswählen (Selectionsort)
- Sortieren durch Einfügen (Insertionsort)
- Sortieren durch Verschmelzen (Beim externen Sortieren)

Die Wahl des richtigen Sortieralgorithmus hängt dabei vor allem von den Anforderungen an den verwendeten Speicherplatz des Algorithmus und der Größe der zu sortierenden Menge ab. Wirth unterscheidet die Sortiermethoden in Bezug auf die Zugriffsmöglichkeiten in zwei Kategorien: [14]

- *Internes Sortieren* (Sortieren von Arrays bzw. Daten im Hauptspeicher)
- *Externes Sortieren* (Sortieren von Daten auf externen Speichermedien)

---

[10] Siehe dazu Kerner, 1990, S. 183.
[11] Ottmann/Widmayer. 1996, S. 23.
[12] Vgl. Hahn/Dzewas, 1991, S. 66.
[13] Vgl. Stahlknecht, 1995, S. 178f.
[14] Vgl. Wirth, 1983, S. 77f.

Die wichtigste Eigenschaft von Sortieralgorithmen besteht in der benötigten Laufzeit des Sortierens. Dabei gibt es speziell in der Sortiergeschwindigkeit große Unterschiede zwischen den Sortieralgorithmen. So kann man die Sortieralgorithmen prinzipiell in relativ langsame, mit einem Laufzeitverhalten proportional zu n^2 (n = Anzahl der zu sortierenden Elemente), und schnelle Algorithmen, mit einem Laufzeitverhalten proportional zu n*log(n), einteilen, wobei keine Methode mit weniger als n*log(n) Vergleichen auskommt.[15] Tabelle 1 zeigt Beispiele für langsame und schnelle Sortieralgorithmen:

Langsame Algorithmen	**Schnelle Algorithmen**
Bubblesort (Austauschen)	Quicksort (Austauschen)
Insertionsort (Einfügen)	Heapsort
Selectionsort (Auswählen)	Mergesort
Shakersort	Bucketsort
Shellsort	

Tabelle 1: Langsame und schnelle Sortieralgorithmen

Prinzipiell läst sich sagen, dass die schnellen Algorithmen meist einen größeren zusätzlichen Speicherbedarf haben (z.B. durch Rekursion, Hilfsfelder). So benötiget man beispielsweise beim Sortieren mit verketteten Listen oder Bäumen n zusätzliche Pointer.

Allgemein ist kein Algorithmus als ungünstig anzusehen, wenn man ihn richtig einsetzt. So ist Bubblesort beim ersten Sortieren eines völlig unsortierten Feldes sehr ungünstig, auch gegenüber Selection- und Insertionsort, besonders bei großen Feldern. Ist das Feld allerdings bereits fast sortiert, so ist Bubblesort doch erheblich schneller. Bei einem bereits sortierten Feld sogar besser als Quicksort oder Heapsort, da bei diesen ein sortiertes Feld teilweise zwischenzeitlich sogar wieder unsortiert vorliegt.
Bei kleinen Mengen ist der Geschwindigkeitsvorteil von z.B. Quicksort gegenüber Bubblesort, Insertionsort oder Selectionsort vernachlässigbar klein, wobei sich dabei der eher einfache Aufbau dieser Algorithmen auszeichnet und für Schüler zum Nachvollziehen gut eignet. Bei großen Mengen ist allerdings klar Quicksort der beste Algorithmus zum Sortieren eines Feldes, der aber das nicht triviale Verständnis vom Aufbau rekursiver Funktionen voraussetzt.

[15] Vgl. Kerner, 1990, S. 184ff und Wirth, 1983, S. 80ff.

2.4.Die Idee des Software Engineering

Der oft betrachtete Nachteil, wenn Anfänger bzw. Schüler Programme entwickeln, besteht in der fehlenden Übersichtlichkeit und Nachvollziehbarkeit des Sourcecode. Dies war bereits ein Problem der traditionellen Programmentwicklung. Der Programmierer begann ein zumeist relativ offen formuliertes Problem aufgrund seiner Erfahrung und spontanen Ideen in einer Programmiersprache zu lösen, ohne sich ein Gesamtkonzept dafür zu überlegen. Eine solche stückweise und unsystematische Vorgehensweise birgt die Gefahr, dass Sonderfälle oder später entdeckte Datenkonstellationen nachträglich im Programmcode (meist durch eine Fülle von Sprungbefehlen) berücksichtigt werden müssen, der dann nur noch schwer zu überblicken ist. Doch gerade die leichte Nachvollziehbarkeit der Programmstruktur ist ein bedeutender Aspekt für das Testen und die Wartung.[16]
Aus diesen Überlegungen erwachsen Anforderungen, die an Programme gestellt werden müssen: Zuverlässigkeit und Vollständigkeit, Verständlichkeit, Modifizierbarkeit bzw. Wartungsfreundlichkeit, Benutzerkomfort und Effizienz. Diese lassen sich dadurch beachten, indem allgemeine Grundsätze beim Programmieren wie Modularität, Lokalität, Uniformität und Selbstdokumentation berücksichtigt werden. Hierbei kommt dem Prinzip der Modularität eine zentrale Bedeutung zu.[17] Unter *Modularisierung* versteht man in der Programmentwicklung die Zerlegung des Gesamtprogramms (Gesamtproblem) in Unterprogramme (Teilprobleme), welche syntaktisch unabhängig voneinander sind. Ein Modul kann von außen nur über genau festgelegte Schnittstellen angesprochen werden und bildet eine Art *Black Box* (Geheimnisprinzip). Der Vorteil einer solchen Arbeitsweise liegt darin, dass Module leicht ausgetauscht, modifiziert oder ergänzt werden können, ohne das gesamte Programm neu zu konzipieren. Ebenso als Strukturierungsinstrument hat sich die Methode der *schrittweisen Verfeinerung* bewährt, bei der das Problem solange in Teilprobleme aufgespalten wird, bis diese direkt in Befehle der verwendeten Programmiersprache codiert werden können. Dies sorgt für eine bessere Überschaubarkeit und vereinfachte Testbarkeit, indem jedes Modul für sich getestet werden kann.[18]
Eine klassische Methode zur Planung und Kontrolle bei der Softwareentwicklung ist das Phasenschema, bei dem die zeitlichen Abschnitte beschrieben werden. Aus dem Software-Lebenszyklus heraus hat sich das Forward Engineering (Wasserfallmodell) (Abb. 1) entwickelt, bei dem die einzelnen Phasen zwar starr aufeinander folgen und Änderungswünsche erst im Nachhinein festgestellt werden können, aber für den

---

[16] Vgl. Kurbel, 1990, S. 46f.
[17] Vgl. ebd., S. 47ff.
[18] Vgl. Diemer, 1992, S. 212, Stahlknecht, 1995, S. 218f und Bleßmann u.a., 2000, S. 15f.

Programmieranfänger einen guten Einstieg in die Planungsarbeit liefert. Die Richtung der Entwicklungsstufen verläuft hierarchisch von oben nach unten (Top-Down-Konzeption).[19] Dieses einfache Modell wurde aufgrund von Starrheit um eine iterative Komponente erweitert, indem in jeder Phase die Möglichkeit gegeben wird, die zurückliegende Phasen zu modifizieren.[20]

Um das Vorgehen des Forward Engineering, also den Weg von der ersten Idee bis zum fertigen Programm und der notwendigen Dokumentation nach zu empfinden, eignen sich besonders gut Sortieralgorithmen.[21]

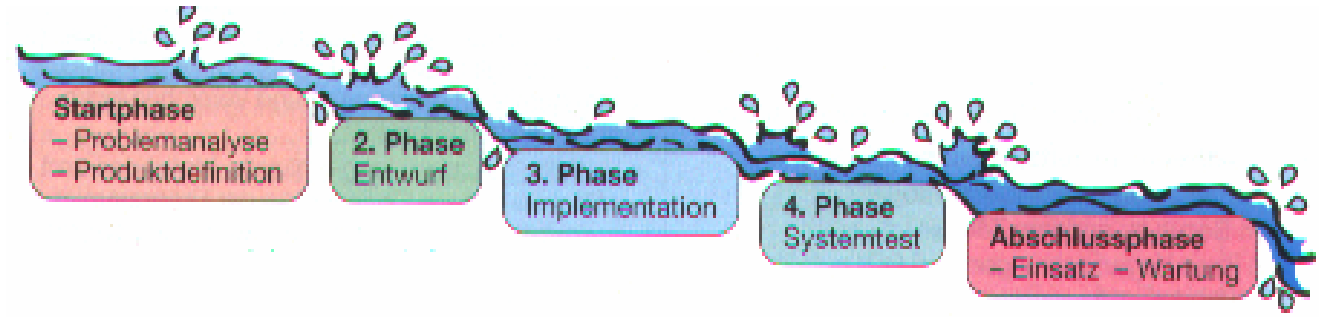

Abb. 1: Klassisches Phasenmodell (Wasserfallmodell)[22]

[19] Vgl. Bleßmann u.a., 2000, S. 12ff.
[20] Vgl. Six u.a., 2004, S. 23.
[21] Vgl. Wolf, 2000, http://www.bildung.hessen.de/abereich/inform/skii/material/delphi/heapsort/
[22] Hübscher u.a., 2001, S. 203.

3. Didaktisch-methodische Orientierung

3.1. Handlungsorientierung und Projektmethode

Die Zielvorstellungen, die aus dem Konzept des Software Engineering resultieren, sind an ganzheitliche inhaltliche und methodische Konzepte gebunden, die wegen ihrer Exemplarität und der Möglichkeit von Eigentätigkeit, bezogen auf das (schulische) Lernen eine hohe Transferfähigkeit sichern und die Handlungsfähigkeit erweitern. Damit rückt die *handlungsorientierte Unterrichtsmethode* ins Blickfeld meiner didaktisch-methodischen Überlegungen zur Umsetzung der Leitziele.

Nach Meyer ist handlungsorientierter Unterricht „ein ganzheitlicher und schüleraktiver Unterricht, in dem die zwischen dem Lehrer und den Schülern vereinbarten Handlungsprodukte die Organisation des Unterrichtsprozesses leiten, so dass Kopf- und Handarbeit der Schüler in ein ausgewogenes Verhältnis zueinander gebracht werden“.[23]

Das Handlungsprodukt kennzeichnet sich dadurch, dass sich der Schüler über seine Herstellung die fachlich relevanten Lerninhalte selbst erschließt. Unter einem Handlungsprodukt verstehe ich „die veröffentlichungsfähigen materiellen und geistigen Ergebnisse der Unterrichtarbeit“[24], die sinnlich wahrnehmbar sind, wie z.B. eine schriftliche Dokumentation, einen Vortrag, eine Präsentation, eine CD, entwickelte Software. Lernfördernd wirkt dabei, dass die Lernenden bei diesem Prozess den Lernstoff zur Lösung ihres spezifischen Handlungsproblems umstrukturieren müssen und sich dadurch aktiv aneignen.

Fast immer wird mit dem Konzept der Handlungsorientierung der Begriff der Projektmethode genannt. Im Unterschied zu anderen Schulfächern, wo das Projekt bzw. der projektorientierte Unterricht als methodische Großform fungiert, ist der Projekt-Begriff innerhalb der Informatik auch wissenschaftlich als Methode verankert. Aus dem traditionellen Verständnis, dass sich projektartige Aktivitäten bei der Softwareentwicklung anhand des Software Life Cycle (s. Kap. 2.4) konkretisieren, scheint der Informatikunterricht für die Projektmethode besonders geeignet. „Platt formuliert: Weil Informatiker projektartig arbeiten, gehört Projektarbeit auch in den Informatikunterricht.“[25]

Diese Sichtweise vermischt allerdings den pädagogischen und den informatischen Projektbegriff: In der Informatik ist das Projekt leistungsorientiert ausgerichtet, in der Pädagogik steht der Arbeits- und Lernprozess, d.h. die Fähigkeit zu kooperativem Arbeiten und Konfliktlösung, ebenso im Mittelpunkt wie das Produkt. Diese unterschiedlichen

---

[23] Meyer, 1991, S. 402.
[24] Meyer, 1991, S. 158.
[25] Thomas, 2003, S. 39.

Auffassungen des Projektbegriffs konkretisieren sich auch in der Rolle der Teamarbeit: Während die Teammitglieder in pädagogischen Projekten für die gesamte Projektlaufzeit konsequent kooperieren, erledigen die Mitglieder informatischer Projekte ihre Aufgabe über weite Strecken unabhängig voneinander in Einzelarbeit. Dadurch behalten die Teilnehmer an pädagogischen Projekten stets den Überblick über das Gesamtprojekt und können ihren eigenen Beitrag einordnen. Eines der wichtigsten Lernziele des Projektunterrichts ist jedoch die Fähigkeit zu kooperativer Arbeit und zur Konfliktlösung in Gruppen. Es wird daher eine Teamstruktur favorisiert, in der alle Teammitglieder annähernd gleichberechtigt sind und die möglichst über die gesamte Projektlaufzeit zwischen allen Teammitgliedern und den Teammitgliedern andere Teams Berührungspunkte (Zwang zur Kommunikation, gemeinsame Ziele) erhält. Dies lässt sich in einem Informatikprojekt umsetzen, indem beispielsweise einzelnen Phasen des Software Life Cycle unterschiedliche Sozialformen zugeordnet werden (z.B. Problemanalyse im Klassenverband, Implementierung in Einzelarbeit) und die Schüler im Wechsel unterschiedliche Aufgaben übernehmen (Protokollführung, Schnittstellenüberwachung, Dokumentation, Präsentation).[26]

Einerseits kann es sich für den Lernprozess unterstützend auswirken, wenn bei der Bildung der Teams das Prinzip der *Differenzierung,* d.h. die Auflösung des heterogenen Klassenverbandes in homogene Gruppen z.B. in Bezug auf die Leistungsfähigkeit, die Vorkenntnisse oder die Interessensrichtung der Schüler, beachtet wird. Andererseits können aber gerade schwächere Schüler in leistungsheterogenen Kleingruppen von den fachlich stärkeren Gruppenmitgliedern profitieren. Maßgeblich wirkt bei der zu treffenden Entscheidung der Schwierigkeitsgrad der Aufgabenstellung. Sehr effizient erweist sich die Differenzierungsmöglichkeit, am gleichen Problem unterschiedliche Anforderungsniveaus (z.B. einfache und schwierige Sortierverfahren) verwirklichen zu lassen (Prototyping).[27]

3.2. Spiralprinzip und Lernschleife

Das handlungsorientierte Lernen erfolgt nach dem Leitgedanken der Prozessorientierung in sogenannten Lernschleifen (siehe Abb. 2), welche durch grob strukturierte Arbeitsaufträge initiiert werden, die Raum für selbstorganisierte Lernprozesse ermöglichen. Orientiert am Arbeitsauftrag stellen die Schüler ein konkretes Produkt bzw. Ergebnis her, präsentieren es und setzen sich in der abschließenden Besprechungssituation kritisch mit diesem auseinander. Diese bildet den Ausgangspunkt für weitere Lernschritte, die die Lernenden

[26] Vgl. Baumann, 1996, S. 183ff und Hubwieser, 2004, S. 71f.
[27] Vgl. Thomas, 2003, S. 40

mit Hilfe des Lehrers planen und erneut durchführen. Der Lehrer übernimmt hierbei stärker die Funktion eines *Moderators* und *Beraters*[28].

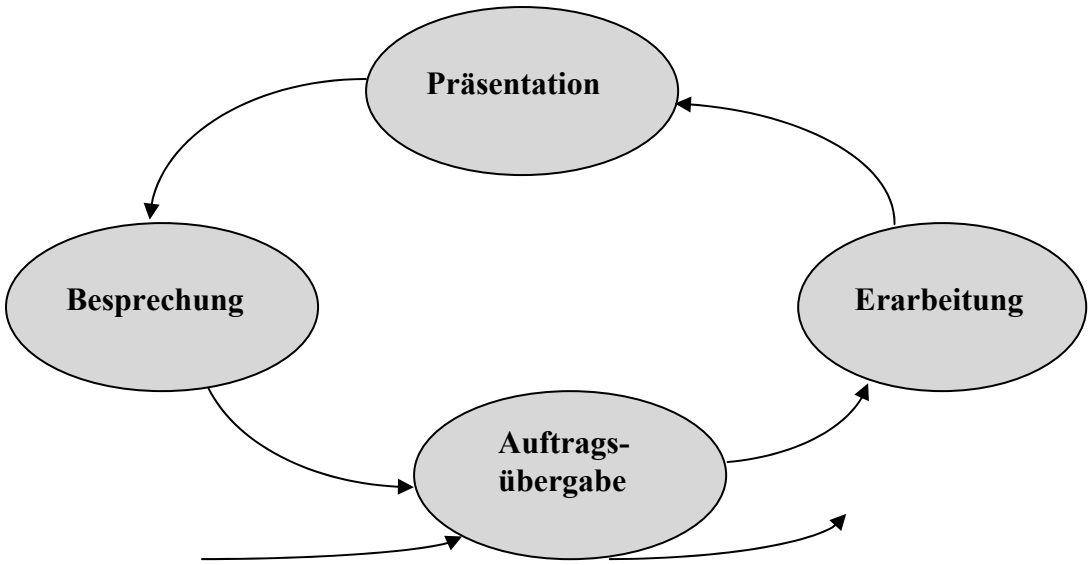

Abb. 2: Die Lernschleife im handlungsorientierten Unterricht[29]

Die Idee der Lernschleife als methodisches Instrument basiert auf dem didaktischen (bzw. curricularen) Ansatz des Spiralprinzips, welches Bruner im Zusammenhang mit der Lehre *fundmentaler Ideen*[30] beschreibt. Dabei soll das Curriculum und die im Unterricht behandelten Inhalte auf diesen grundlegenden Ideen eines Faches aufbauen und wiederholt in anderen Zusammenhängen bzw. auf höheren Abstraktionsniveaus aufgegriffen werden.[31] Schwill analysiert aufgrund der Ausführungen von Bruner drei Teilprinzipien des Spiralprinzips heraus:[32]

- **Prinzip der Fortsetzbarkeit**
  (Ausbau eines Themas auf verschiedenen Niveaus)
- **Prinzip der Präfiguration von Begriffen**
  (Erklärung eines Begriffs auf verschiedenen Darstellungsstufen)
- **Prinzip der vorwegnehmenden Lernens**
  (Behandlung eines Themas auf einfachen Stufen und nicht endgültig)

---

[28] Vgl. Mäder, 1997, S. 34.
[29] Vgl. Arnold/Müller, 1993, S. 327.
[30] Vgl. Schwill, 1996, S. 17ff und Baumann, 1996, S. 51ff.
[31] Vgl. Bruner, 1960, S. 26f.
[32] Vgl. Schwill, 1996, S.22f.

Für die Umsetzung im Unterricht entwickeln Lisop/Huisinga aus der Idee des Spiralprinzips heraus drei Phasen des Spiralverlaufs:[33]

(1) Phase der Konfrontation
(Erkennen des Problems)

(2) Phase der Reorganisation
(Rückbezug auf Gelerntes, Entwickeln einer Lösung, Verknüpfung des neuen zum Bekannten)

(3) Phase der Anwendung und Übung
Übernahme neuer Denk- und Handlungsmuster)

Aufgrund der dargestellten Überlegungen kann man konstatieren, dass sich Lernschleife und Projektmethode methodisch gut ergänzen, indem soziale und kommunikative Kompetenzen vermitteln werden. Bei der Umsetzung der fächerübergreifenden didaktischen Ansätze werden Beziehungsnetze zwischen den Inhalten geschaffen, die durch ihre längerfristige Gültigkeit Sinn stiften und dadurch besser anzuwenden sind.

3.3. Informatische Trias

Es haben sich seit der Einführung des Fach Informatik unterschiedliche fachspezifische didaktisch-methodische Ansätze herausgebildet, die heute weniger rein dogmatisch nebeneinander existieren.[34]
Informatikunterricht soll nach Hubwieser so gestaltet werden, dass außer der Umsetzung von Lernprozessen (Informatik als Lern- und Informationshilfe) und Bedienerschulung (Umgang mit Informatiksystemen) vor allem das Problemlösen (Modellieren, Strukturieren und Bewerten) unter Anwendung von grundlegenden Informatikprinzipien und -methoden thematisiert werden. Natürlich können die drei Bereiche nicht losgelöst voneinander inhaltlich behandelt werden, da sie sich gegenseitig bedingen. Daraus resultiert für die konkrete Gestaltung von Unterricht, dass die informatische Trias, d.h. der Implikationszusammenhang von Lernhilfe, Bedienerfertigkeit und die Vermittlung grundlegender Konzepte (fundamentaler Ideen), als Einheit betrachtet werden (siehe Abb. 3). [35]

---

[33] Vgl. Lisop/Huisinga, 1994, S. 212ff.
[34] Vgl. Baumann, 1996, S. 112ff.
[35] Vgl. Hubwieser, 2004, S. 43ff.

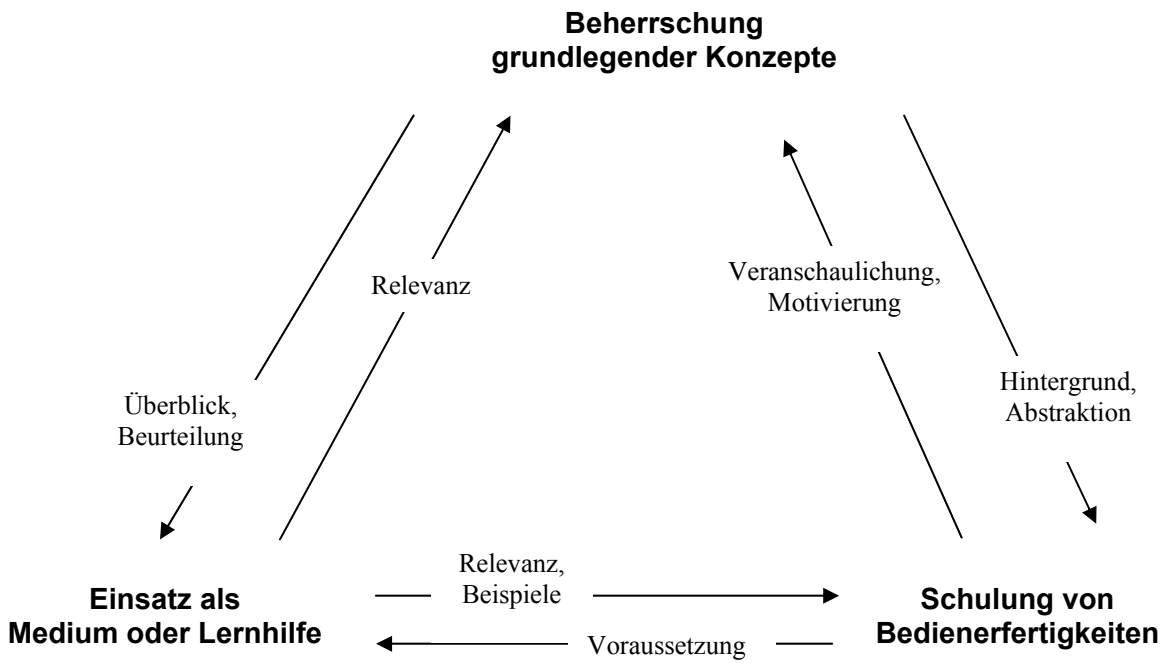

Abb. 3: Verflechtung von Unterrichtshilfen, Bedienerschulung und Vermittlung grundlegender Konzepte[36]

Magenheim abstrahiert in seinen Ausführungen zur systemorientierten Didaktik von diesen 3 Bereichen, indem er die zentrale Aufgabe der Informatik in der Gestaltung von Informatiksystemen sieht. Unter einem Informatiksystem versteht er die Einheit von Software, Hardware und sozialen Handlungssystemen von Personen, die mit dem technischen Teil und miteinander interagieren.[37]

Zur Umsetzung dieser Konzepte eignet sich der systemanalytische Zugang von Schwill, bei dem die Schüler in einem top-down-orientierten Vorgehen (Blick auf das System, Blick in das System, Modifikation des Systems, Konstruktion des Systems) ein komplexes Softwaresystem studieren.[38] Auch Breier befürwortet als methodische Vorgehensweise eine problembezogene Orientierung am Software-Lebenszyklus (Problembegegnung, Problemanalyse, Modellbildung, Algorithmierung, Programmierung, Implementierung, Modellkritik, Rückbesinnung, Weiterführung) mit einer begleitenden Dokumentation. Er verlangt dafür ein ausgewogenes Verhältnis zwischen computerferner Arbeit und der Implementierung auf dem Computer.[39]

---

[36] ebd., S. 49.
[37] Vgl. Magenheim, 2003, S. 2ff.
[38] Vgl. Schwill, 1996, S. 51.
[39] Vgl. Breier, o.J., S. 4ff.

4. Unterrichtspraktische Umsetzung

4.1. Lehrbedingungen

4.1.1. Rechtliche Einordnung und Vorgaben

Das Fach Datenverarbeitung im beruflichen Gymnasium mit der Fachrichtung Wirtschaft und Verwaltung ist ein 3-Stündiges Pflichtfach in der Jahrgangsstufe 11 und muss in der Qualifikationsphase mindestens mit 2 Kursen je 40 Wochenstunden belegt sein, wobei nur ein Kurs in die Wertung des Grundkursbereichs eingebracht werden braucht.[40]

Die rechtlichen Grundlagen für die Auswahl der Inhalte im Unterricht und den Abiturprüfungen bilden die einheitlichen Prüfungsanforderungen in der Abiturprüfung (EPA-DV)[41] von 1989, der Rahmenlehrplan[42] (RLP) von 1985 und die Fachspezifischen Prüfungsanforderungen (FAPA)[43] von 2003.

Die EPA legen einen Rahmen fest, der in den FAPA vollständig übernommen wird, welche Inhalte im Unterricht der Oberstufe vermittelt werden sollen. Dabei wird unterschieden nach Grundlagen der strukturierten Programmierung (Problemanalyse, Datenstrukturen, Ablaufstrukturen, Programmentwurf, Codierung, Programmtest, Dokumentation), fachspezifische Anforderungen an Hardware und Software, Auswirkungen der IuK-Techniken und fachrichtungsspezifischen Inhalten (z.B. Phasenmodell, Umgang mit Standardsoftware, Formen der Datenorganisation, Simulationen, Netzwerke). Als didaktische Begründungen werden die Ausrichtung an der Fachsystematik und der anwendungsorientierten Informatik genannt.

Bei der Konkretisierung in den Lehrplänen findet man als weitere didaktische Begründungen die Problemlösungs- und Entscheidungsorientierung, die durch das Konzept *Problem-Algorithmus-Programm-Computer* strukturiertes Vorgehen vermitteln, konstruktives Denken und Selbsttätigkeit fördern. Als fachliche Qualifikation steht in der Jahrgangsstufe 11 die algorithmische Problemlösung mittels einer problemorientierten Sprache im Mittelpunkt und am Schuljahresende soll im geringen Umfang der Umgang mit Computer, Peripherie und betriebssystemtypischen Vorgängen behandelt werden. In der Jahrgangsstufe 12 und 13 steht die Thematisierung der vier Kurse Programmierung (Einübung der algorithmischen Problemlösung), Aufbau und Funktionsweise einer Computeranlage (vertiefte Hardware- und Softwarekenntnisse), Datenbanken und anwendungsbezogene Problemlösungen an, wobei der erst genannte Kurs, in welchen mein Unterrichtprojekt einzuordnen wäre, bezüglich der Reihenfolge obligatorisch ist.

---

[40] Vgl. VOGO/BG, 2003, Anlage 6 zu §16 (1), Anlage 7 zu § 19 (1) und § 26.
[41] Vgl. KMK, EPA-Datenverarbeitung, 1989.
[42] Vgl. HKM, Rahmenlehrpläne Datenverarbeitung, 1985.
[43] Vgl. VOGO/BG, 2003, Anlage 11 (19.1.3.)

4.1.2. Schulische Rahmenbedingungen

Die inhaltliche Rahmenbedingungen, die konkret durch den RLP vorgegeben wird, schien den EDV-Lehrern an unserer Schule im Hinblick auf Anforderungen in der Berufs- und Lebenswelt veraltet und bezogen auf unser Schülerklientel zu demotivierend. Die Fachkonferenz Datenverarbeitung entschied eine Umstrukturierung des Lehrplans unter Beachtung der EPA und FAPA vorzunehmen. Es beziehen sich die ersten zwei Drittel der Unterrichtsinhalte in der Jahrgangsstufe 11 auf die kaufmännische Standardsoftware (Word, PowerPoint, Excel) in Verbindung mit den Themen Betriebssysteme und Hardwarekomponenten unter Einbeziehung des Internets und Beispielen aus Wirtschaft und Mathematik. Jeder Lehrer kann dabei entscheiden, wie intensiv einzelne Themen behandelt werden. Daran schließt sich der Bereich Algorithmierung und Programmierung mit dem *C++ - Builder* von Borland[44] an, welcher bis zum Ende des ersten Halbjahres der Klasse 12 fortgesetzt wird. Im zweiten Halbjahr folgt der Themenkomplex Datenbanken (Theorie und Anwendung mit Access) und im letzen Schuljahr lernen die Schüler SQL sowie je nach Abstimmung mit dem Kursleiter Internetprogrammierung, Java oder VBA.

Die inhaltliche Umsetzung des durchgeführten Projektes Sortieralgorithmen findet in der zweiten Hälfte des ersten Halbjahres der Jahrgangstufe 12 statt, welches der chronologischen Reihenfolge entspricht. Die Schüler haben in der 11. Klasse die Komponenten Button, Edit und Label sowie einiger ihrer Eigenschaften kennen gelernt und sollten in der Lage sein, ein Formular zu gestalten. Als imperative Sprachausdrücke wurden Variablendeklarationen, Rechenopertoren und logische Operatoren, Verzweigungstypen und Schleifen (for, do while, while) behandelt und die Ein- und Ausgabe über die Komponenten besprochen. Zu Beginn der Jahrgangsstufe 12 beschäftigten sich die Schüler im Zusammenhang mit (verschachtelten) Schleifen mit der StringGrid-Komponente und ein- und zweidimensionalen Arrays.

Unterrichtsorganisatorisch wird bei der Gestaltung des DV-Unterrichts darauf geachtet, dass die Kurse nicht mehr als 20 Schüler enthielten. Dadurch hat jeder Schüler die Möglichkeit, mit dem eigenen Account über einen Rechner (=Client des Schulnetzwerkes) zu verfügen. Die Lösungen und Arbeiten von Mitschüler sind bei diesem Verfahren nicht zugänglich. Die Teilnahme an den beiden Grundkursen Datenverarbeitung in der Jahrgangsstufe 12 sind für alle Schüler verpflichtend und finden jeweils in einer Doppelstunde pro Woche statt.

---

[44] Die Wahl dieser Programmiersprache wird damit begründet, dass sie Konzepte enthält, die in anderen Sprachen auch zu finden sind, relativ leicht „office-ähnliche“ Programme (Fenstertechnik) geschrieben werden können und einen Zugang zu einer objektorientierten Sprache ermöglicht.

4.1.3. Die Schüler des Grundkurses Datenverarbeitung

Die 18 Schüler (6 weiblich, 12 männlich) sind zwischen 17 und 20 Jahre alt und wohnen zerstreut im Hochtaunuskreis, Maintaunuskreis und Frankfurt. 10 Schüler sind mir bereits aus dem DV-Unterricht der Jahrgangsstufe 11 bekannt, 5 der unbekannten Schüler belegen gleichzeitig den LK Mathematik bei mir.

Trotz des schulinternen Lehrplans weisen die Schüler der 12. Klasse eine relativ heterogene Vorbildung auf. Nur sechs (männliche) Schüler durchschauen das in Klasse 11 begonnene Thema Programmierung. Einerseits wird das Erlernen einer Programmiersprache von den meisten abgelehnt, andererseits sind manche Inhalte nicht behandelt wurden. Hierbei verdeutlicht sich auch die grundlegende Haltung vieler Schüler gegenüber Programmiersprachen, für welche sich der Computer nur zum „Fun haben“ eignet und es viel zu schwierig erscheint, die Funktionsweise zu erkunden.

Diese Einstellung wirkt sich auf die **Leistungsbereitschaft** aus: Pascal, Andreas, Nicholas, Holger, Carsten, Kai beteiligen sich regelmäßig am Unterricht, wobei Pascal und Andreas aufgrund ihrer Ausführungen den Unterricht dominieren. Sebastian Se., Daniel, Andreas H., Rebecca, Elisa und Pia melden sich selten und sind nur bereit in Zusammenarbeit mit anderen Mitschülern Programmieraufgaben zu lösen. Verena, Peter, Benjamin, Sebastian Sa., Janine und Iva geben auch bei direkter Ansprache kaum eine Antwort und verhalten sich in Gruppenarbeitsphasen eher abwartend. Es hat sich in einigen Unterrichtseinheiten gezeigt, dass die Motivation der Schüler insgesamt zunimmt, wenn Schülerergebnisse (z.B. ausführbare Programme) zum Unterrichtsgegenstand erhoben werden.

Die **Leistungsfähigkeit** ist ebenfalls differenziert zu beurteilen: Während auch die aktiven Schüler, bis auf Nicholas, in ihren mündlich und schriftlichen Leistungen gut sind, weisen die übrigen Schüler eher schwache Ergebnisse auf. Die Abstraktionsfähigkeit der schwächeren Lerngruppe und das Denken in komplexen Zusammenhängen ist grundsätzlich eher eingeschränkt. Das Erkennen eines Problems, dessen Lösung und Umsetzung mittels einer Programmiersprache wird nur langsam und lückenhaft von ihnen vollzogen, wobei die Grundlagen ständig vergessen und im Buch oder Ordner nach geschaut werden.

Bei der **Arbeitshaltung** der Schüler fällt positiv auf, dass sie versuchen sich mit der Aufgabenstellung auseinander zu setzen, auch wenn nur kleine Erfolge sichtbar sind. In der Regel konzipiert jeder Schüler selbst oder in Partnerarbeit das Aussehen seines Formulars auf der GUI, lediglich beim Programmieren wird auf verbale Erklärungen der besseren Schüler bezug genommen, denn ein Kopieren des Quellcodes ist wie oben beschrieben nicht möglich. Diese Arbeitshaltung wirkt sich positiv auf die **Lernatmosphäre** aus, die als harmonisch bezeichnet werden kann. Es haben sich zwar einzelne Grüppchen gebildet,

zwischen denen meines Erachtens keine Konkurrenz besteht, sondern die Leistungen werden untereinander akzeptiert.

4.2. Planung, Durchführung und Reflexion der Unterrichtsprozesse

4.2.1. Einordnung der Leitziele Selbständigkeit, Selbsttätigkeit, Selbstreflexion

Die Schulen und Lehrer sollen nach dem novellierten hessischen Schulgesetzes die Schüler befähigen, „ihr zukünftiges privates, berufliches und öffentliches Leben auszufüllen, bei fortschreitender Veränderung wachsende Anforderungen zu bewältigen und die Freizeit sinnvoll zu nutzen“[45] Als zu vermittelnden Fähigkeiten werden neben den fachlichen Fertigkeiten u.a. genannt, die Fähigkeit der Zusammenarbeit und des sozialen Handelns, Konflikte zu lösen und zu ertragen, aber auch die Fähigkeit, Kreativität und Eigeninitiative zu entwickeln.[46] Zur Realisierung dieses umfassenden Bildungs- und Erziehungsauftrages sind Unterrichtsformen und Methoden anzuwenden, die die Selbständigkeit, Selbsttätigkeit und Selbstreflexion anhand geeigneter Bildungsinhalte fördern. Diese zu vermittelnden Qualifikationen beschreibt Modrow im Zusammenhang mit den Aufgaben und Zielen der Informatikdidaktik als „Lernen in Strukturen“, „Entdeckendes Lernen“ und „Soziales Lernen“ und begründet dies anhand konstruktivistischer Lerntheorien. [47]

Unter **Selbständigkeit** verstehe ich die Kompetenz einer Person, aufgrund einer bestimmten Problemstellung oder Zielsetzung geeignete Mittel und Verfahren zur Lösung eines fachlichen Problems zu kennen und zu finden. Dazu gehören im einzelnen die Problemwahrnehmung, Überlegungen zur Problemlösung und die Planung der Problemlösung durch die Anwendung und Erweiterung bereits erworbener Handlungs- und Deutungsmuster. Die **Selbsttätigkeit** besteht in der konkreten praktischen Umsetzung der gefundenen Mittel und Verfahren und weist auf den Handlungsbezug hin. Die Schüler setzen die Planung um, indem sie etwas “*tun*“: Sie beschaffen sich Informationen, die sie auswerten, aufbereiten und darstellen, sie gestalten die Unterrichtsvorhaben mit, indem sie z.B. im Internet suchen, eine Präsentation durchführen oder ein Projekt gestalten. Die Förderung von Selbsttätigkeit lässt sich lernpsychologisch begründen. Durch selbständiges Handeln erworbene Fähigkeiten und Kenntnisse werden wesentlich besser behalten als durch auditive, visuelle oder audiovisuelle Verfahren.[48] Unter **Selbstreflexion** verstehe ich das individuelle Wahrnehmen, Auslegen und Hinterfragen der realisierten Entscheidungen

[45] HSchG § 2 (2), 1997.
[46] Vgl. ebd. § 2.
[47] Vgl. Modrow, 2002, S.14ff.
[48] Vgl. Freytag, H.-P .u.a., 1992, S. E28.

und Handlungen in bezug auf die Planung, das Thema und die Gruppe bzw. Klasse. Sie dient der Überprüfung der bestehenden Handlungs- und Deutungsmuster, führt ggf. zu ihrer Reorganisation und erweitert somit die individuelle, soziale und berufliche Handlungsfähigkeit bzw. Selbständigkeit. Gerade die Selbstreflexion ermöglicht den Schüler, ihren Platz in der ihn umgebenden Welt (selbst)bewusst, eigenverantwortlich und selbstbestimmt einzunehmen.

4.2.2 Ablaufstruktur

Das Unterrichtsprojekt *Sortierverfahren* durchläuft zwei Schleifen des Spiralprinzips: Innerhalb der ersten Schleife (Unterrichtsprozess I) erfolgt die Problemanalyse und Lösungsverfahren zu der Aufgabenstellung, die zweite Schleife (Unterrichtsprozess II) legt ihren Schwerpunkt auf Entwurf, Implementation und Test. Jede Schleife besteht aus den drei Phasen Konfrontation, Reorganisation und Anwendung (siehe Abb. 4).[49]

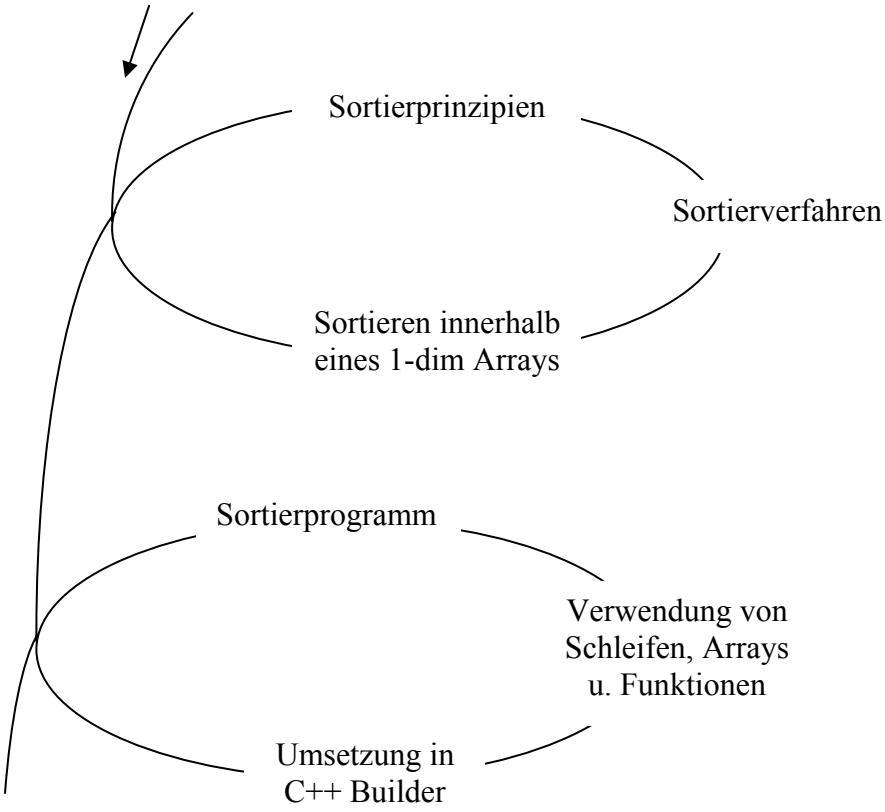

Abb. 4: Phasen des Spiralverlaufs zum Unterrichtsprojekt „Sortierverfahren“

[49] Vgl. Kap. 3.2.

Beide Schleifen zusammen betrachtet spiegeln die Vorgehensweisen nach dem Wasserfallmodell wieder (s. Abb. 5). Die Abbildung 5 verdeutlicht zudem, welche Produkte die Schüler nach der Projektmethode erstellen sollen.

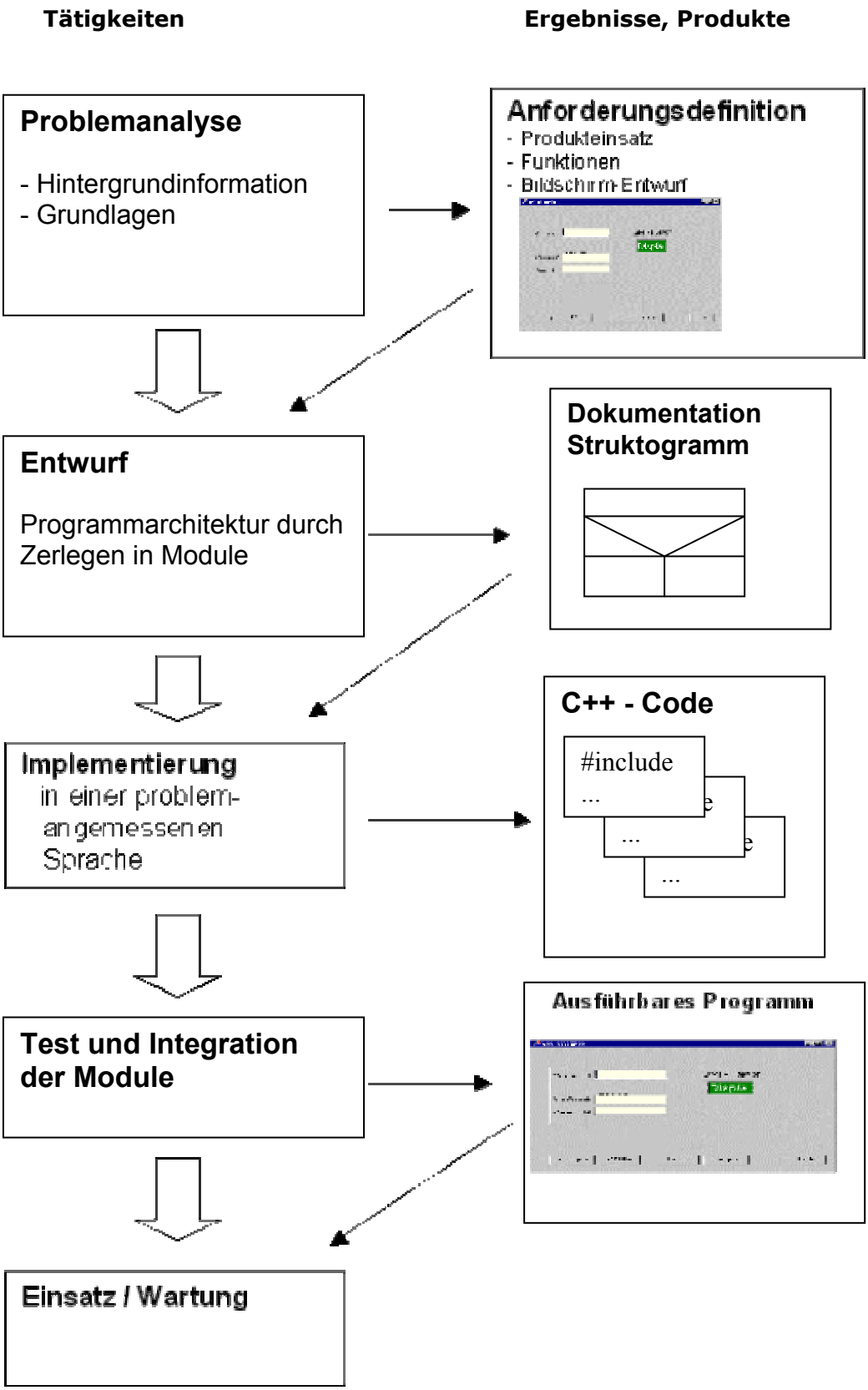

Abb. 5: Vorgehensweise nach dem Wasserfallmodell im Unterrichtsprojekt

Methodisch soll nach der Lernschleife in Verbindung mit der Projektmethode vorgegangen werden, um die Realisierung der Leitziele zu unterstützen. Als zeitlichen Rahmen sind 4-6 Wochenstunden (14.11.-28.11.) für den Unterrichtsprozess I und 6-8 Wochenstunden (28.11.-19.12) für den Unterrichtsprozess II geplant.

4.2.3 Unterrichtsprozess I (Problemanalyse, Lösungsverfahren)

4.2.3.1. Lernziele

Fachliche Lernziele (FL)

Die Schüler sollen durch eine kurze Darstellung des Lehrers am Beispiel des Sortieren bzw. Auswertens von Lottozahlen (6 aus 49) einen Einblick in die Thematik gewinnen und erörtern, dass Sortiervorgänge die spätere Suche vor allem in großen Datenbeständen wesentlich vereinfachen und somit zur Zeit- und Kostenersparnis beitragen. Zudem soll die Lerngruppe aufdecken, dass die Lottozahlen innerhalb eines Programms in Form eines eindimensionalen Arrays vorliegen und diese Anordnung beibehalten werden muss. Die Schülen müssen differenzieren zwischen Anordnung und Reihenfolge der Zahlen.

Im nächsten Unterrichtsabschnitt besteht ein weiteres Ziel darin, dass die Schüler verschiedene Lösungen des Sortierens kennen lernen und den Sortierprinzipien einige Sortierverfahren (Bubblesort, Selectionsort, Insertionsort und Quicksort) zuordnen können. Für das Ziel, die Beherrschung der grundlegenden Konzepte[50], ist es daher notwendig, dass die unterschiedlichen Prinzipien des internen Sortierens dargestellt und beschrieben werden können. Beim *Sortieren durch Austausch* soll aufgezeigt werden, dass jeweils benachbarte Zahlen miteinander verglichen und so lange ausgetauscht werden, bis die gewünschte Reihenfolge erreicht ist. Zudem soll zu diesem Prinzip anhand der prinzipiellen Funktionsweise des Quicksort-Verfahren die Gemeinsamkeiten (Vergleich von 2 Zahlen und Tausch) und Unterschiede (rekursives Teilen) herausgestellt werden. Die Schüler sollen ermitteln, dass während des *Sortierens durch Auswählen* aus der ursprünglichen Zahlenanordnung die kleinste Zahl gesucht wird und an den Beginn der neuen Zahlenreihenfolge gestellt wird. In der verbleibenden ursprünglichen Zahlenanordnung wird wieder die kleinste Zahl gesucht und an die zweite Stelle der neuen Reihenfolge gesetzt. Dieses Prinzip setzt sich weiter fort. Bei *Sortieren durch Einfügen* soll verdeutlicht werden, dass nacheinander jede Zahl der alten Zahlenanordnung in einer neu anzulegenden Reihenfolge einzeln an die richtige Stelle der Sortierung gebracht wird, indem die Zahl paarweise mit den bereits eingeordneten Zahlen verglichen wird.

Als weiteres Ziel sollen die Schüler die Vorgehensweise zur Auswertung der Gewinnklassen des Lottospiels nennen.

---

[50] Vgl. Kap. 3.3.

Lernziele, die die Selbständigkeit und Selbsttätigkeit fördern (LSS)

Während der Projektarbeit sollen die Schüler in einem Team Aufgaben eigenständig aufteilen (Koordinationsfähigkeit) und Verantwortung übernehmen. Eigene Verantwortung übernehmen bedeutet aber auch immer soziale Verantwortung zu übernehmen, d.h. der Schüler soll seine Ergebnisse, Kenntnisse und sein Wissen mit den anderen Gruppenmitgliedern teilen und schwächere Schüler unterstützen (Kooperationsfähigkeit). Die Schüler sollen üben in der Kleingruppe und während Präsentationen ihren Mitschülern zuzuhören und sachlich zu diskutieren (Kommunikations- und Argumentationsfähigkeit).

Die Schüler entwickeln die Fähigkeit zum selbständigen Problemlösen und eigenen Strukturieren unter Benutzung des Computers als Medium und Lernhilfe (Internet). Ein weiteres methodisches Lernziel besteht in der Schulung von Bedienerfertigkeiten, z.B. der Umgang mit Standardsoftware, Betriebssystem und Speichermedien. Die Schüler sollen ihre Arbeitsphasen dokumentieren und die Gruppenergebnisse im Plenum präsentieren, d.h. ihre Lösung vorstellen.

Lernziele, die die Selbstreflexion fördern (LS)

Ein Lernziel besteht in der Erfahrung und dem Umgang von (fachlicher) Kritik an den eigenen Ergebnissen (Konfliktfähigkeit und Ambiguitätstoleranz), indem die Schüler Antworten zu den gestellten (Verständnis)Fragen geben. Auf diesem Weg sollen sie sich die Fähigkeit zum Argumentieren auf unterschiedlichen Präzisierungsniveaus (erkennen - lösen - wissen - bewerten) aneignen und Offenheit gegenüber Problemsituationen entwickeln. Sie sollen erkennen, das reale Probleme mit Hilfe der Informatik effizient dargestellt und gelöst werden können, aber Lösungen nicht eindeutig sind und hinterfragt werden können.

4.2.3.2. Planung, Durchführung und Reflexion[51]

Planung

Im Vordergrund meiner Überlegungen steht die aktive Beteiligung der Schüler bei ihrer Aneignung neuer Konzepte, denn die Möglichkeit zur Mitgestaltung scheint mir ein plausibler Weg hin zu eigenverantwortlichen Lernen und Arbeiten zu sein.

[51] Anm.: Programmbeispiel (Quellcode oder Formular), Arbeits- und Übungsblatt befinden sich im Anhang.

Geplanter Verlauf des Unterrichtsprozess I

Phase	Geplanter Verlauf	Sozial-Aktionsform	Medien/ Materialien
Einstieg	L konfrontiert S mit Thema und stellt Ablauf des Projektes vor	-L-Vortrag	-PC + Beamer -Programmbsp.
Erarbeitung	S bilden Gruppen und teilen Module bzw. Sortierverfahren untereinander auf Bearbeitung der Aufgaben Aneignung der Grundlagen und Hintergrundinformation Vorbereitung der Präsentation	-Gruppenarbeit -L berät	-Arbeitsblatt -Internet -Schulbuch -Standard-software
Präsentation	Jede Gruppe stellt ihr Sortierprinzipien/-Verfahren bzw. Verfahren für die Gewinnklassen dar, erklärt und antwortet auf Fragen.	-Plenum -S-Vortrag -S-S-Gespräch -L moderiert	-PC + Beamer -Tafel -Plakate
Reflexion	Sortierprinzipien/-verfahren werden verglichen Diskussion um Güte der Sortierverfahren	-S-L-Gespräch	-Tafel
Ergebnis-sicherung	Schüler bearbeiten Übungen und vergleichen gemeinsam im Unterricht	-Einzelarbeit	-Übungsblatt

Um den Schüler eine Vorstellung von der zu lösenden Aufgabe zu geben, stelle ich ihnen ein mögliches Programmbeispiel vor, das noch nicht ausgereift ist. Ich hoffe auf eine Motivation der Schüler, das Spiel ebenfalls zu entwickeln, denn beim Zeigen der Ziehung wird deutlich, wie unwahrscheinlich es ist, selbst mit nur drei Richtigen zu gewinnen. Im weiteren Gespräch soll die Relevanz von Sortiervorgängen deutlich werden. Ich habe ausgehend von Vorkenntnissen der Schüler das Sortieren von Lottozahlen gewählt, da Zeichenketten noch nicht im Unterricht behandelt wurden, beim Testen des Programms die Korrektheit leicht nachgeprüft werden kann und es ein bekanntes Spiel ist. Daran anschließend bespreche ich anhand des Arbeitsblattes den weiteren Ablauf des Projektes und organisiere die Gruppenbildung (6 Gruppen mit je 3 Schülern). Meine Intention besteht darin, dass die guten Schüler sich getrennt in den Gruppen befinden, denn sonst befürchte ich aufgrund der Leistungsfähigkeit der meisten Schüler, dass das Gesamtprogramm bestehend aus 3 Modulen nicht erstellt wird.

In der Erarbeitungsphase, die selbständig und selbsttätig durchgeführt wird, eignen sich die Schüler in der Regel mit Hilfe von Internetquellen das Wissen des jeweiligen Sortierprinzip bzw. -verfahren und die Gewinnklassenermittlung an. Hierbei wird, als Phasen des

Spiralprinzips, der anschauliche Bezug zum bereits gehandelten eindimensionalen Array hergestellt und das zugrundeliegende informatische Konzept erkannt.

Dieses neue Wissen festigt sich in der Präsentation, denn es findet die inhaltliche Auseinandersetzung in einem schwierigeren sozialen Rahmen (vor der Gruppe) statt. Nur wer das Prinzip vollständig durchschaut hat, kann sicher auf die Fragen und anderen Vorträge der Mitschüler eingehen. Die Schüler werden bestrebt sein, ihre Ergebnisse gut vorzutragen, um sich nicht die Blöße des Versagens zu geben. Durch die fachliche Festigung kann eine Reflexion positiv gestaltet und erweiternde Fragestellungen behandelt werden.

Die Ergebnissicherung dient den Schüler vor allem dazu, dass bei der Vorstellung der Programme im Unterrichtsprozess II die dahinter stehenden Prinzipien schnell rekapituliert werden können. Ein erweiternder Aspekt der Übungsaufgaben bezieht sich auf die mögliche Thematisierung der Laufzeit (Komplexität) im Unterricht.

Alle Phasen sollen von den Schülern dokumentiert werden. Dies hilft, leichter einen Einstiegspunkt in den nächstfolgenden Stunden zu finden sowie ermittelte Zwischenergebnisse und Quellen festzuhalten.

Durchführung

Die Idee des Projektvorhaben nahmen die meisten Schüler sehr positiv auf und fanden die Idee des Lottospiels mit automatischer Ausgabe der Gewinnklassen interessant. Kritisiert wurde allerdings die Sortierung der Tippeingabe, da dies „realitätsfremd“ sei. Trotz dieses kleinen „Makels“ ließ sich die Gruppe auf das Projektvorhaben ein, denn der Sinn des Sortierens, „das schnelle Finden der Daten“, welches die Schüler nannten, wurde als wesentlich bewertet. (FL+)[52]

Die Gruppenbildung gestaltete sich einfach, denn die Anweisung, die guten Schüler auf die Gruppen der unterschiedlichen Sortierverfahren zu verteilen, wurde anerkannt. Bei der Gruppenzusammensetzung zum Thema Quicksort verhielt ich mich offen, weil die dazu notwendigen Inhalte (Funktion, Rekursion) im Unterricht nicht behandelt wurden und eher von den leistungsstarken Schülern selbständig nachgearbeitet werden können. Es entstanden folgende Gruppen:

Gruppe A1 Gewinnklassen	Gruppe A2 Quicksort	Gruppe A3 Bubblesort	Gruppe B1 Gewinnklassen	Gruppe B2 Insertsort	Gruppe B3 Selectsort
Nicholas, Pia, Verena,	Holger, Carsten, Andreas H.	Andreas, Peter Janine	Sebastian, Iva, Sebastian	Pascal, Elisa, Benjamin	Kai, Daniel, Rebecca

[52] + (-) bedeutet, die Durchführung, Handlung bzw. Darstellung der S erfüllt das angegebene Lernziel eher (eher nicht).

Innerhalb der Gruppen verlief die Arbeitsaufteilung reibungsfrei, und zügig wurden die Rollen des Projektleiters, Projektassistenten oder Dokumenteurs zugewiesen. Die Arbeitshaltung in den Gruppen vermittelte mir den Eindruck, dass die Schüler konzentriert und engagiert arbeiteten, denn es wurden viele Internetseiten zum Thema Sortierverfahren gelesen, Verständnisprobleme untereinander geklärt und Lösungen diskutiert. (LSS+, LS+)

Zu Beginn der dritten Doppelstunde gaben die Schüler ein kurzes Feedback ihrer Vorgehensweise bzw. Gruppenarbeit und stellten die einzelnen Sortierverfahren im Plenum vor. Obwohl die Vortragenden Andreas, Kai, Holger und Carsten als Medium nur die Tafel benutzten, erläuterten alle sehr anschaulich und korrekt die Sortierprinzipien mit Hilfe von Zahlenreihen (=Anordnungen) und Pfeilen. Pascal benutzte zur Veranschaulichung ein zurechtgeschnittenes Stück Pappe und kleine Karteikarten, um das Sortieren durch Einfügen darzustellen (siehe Anhang). (FL+) Es fand eine produktive inhaltliche Auseinandersetzung mit den vorgestellten Prinzipien statt und die Zuordnung zu den Sortierverfahren wurde richtig genannt. Die Vortragenden erwiesen sich als wahre Spezialisten, denn sie konnten auf Fragen der Mitschüler sehr gut antworten und waren kritischen Anmerkungen gegenüber aufgeschlossen. (LSS+, LS+). Bei der Darstellung der Gewinnklassenermittlung lieferte Nicholas einen durchschnittlichen, Sebastian Sauerbrei einen sehr schlechten Beitrag ab. (FL+-) Es fiel negativ auf, dass sich die Schüler dieser beiden Gruppen an der Diskussion zu den Sortierverfahren nicht beteiligten. (LS-)

Aufgrund der guten Arbeitsatmosphäre stellte ich die Frage nach dem „besten Verfahren“ stellen. Zuerst musste die Frage geklärt werden, was „besser“ eigentlich bedeutet. Andreas und Pascal bezogen sich richtigerweise auf die Laufzeit und damit auf die durchzuführenden Vergleich- und Tauschvorgänge. Für die anderen Schüler wurde dieses Themengebiet zu abstrakt und sie schienen mir überfordert, denn mit Beginn der rechnerischen Ermittlung der Komplexität beim Sortieren durch Austausch (Bubblesort) wurde es im Klassenraum unruhig. Da dieses Thema zudem nicht zum curricularen Pflichtbereich zugeordnet ist, brach ich meine Ausführungen ab. (LS-)

Das Übungsblatt gab ich als Hausaufgaben mit, da die Präsentations- und Reflexionsphase mit einer Doppelstunde länger dauerten, als ich dachte, und besprach die Ergebnisse in der darauffolgenden Stunde.

Reflexion

Der Unterrichtsprozess I verlief sehr positiv, denn mit dem Erreichen der intendierten fachlichen sowie überfachlichen Lernziele wurde gleichzeitig das Konzept der informatischen Trias umgesetzt. Die Schüler arbeiteten viel engagierter als im lehrerzentrierten Unterricht bzw. bei der Einzelarbeit mit. Es war ihnen wichtig, dass ihre

dargebotenen Erklärungen die Mitschüler verstanden. Aus diesem Grund bereiteten sich die Vortragenden gewissenhaft vor, und hatten durch die inhaltliche Auseinandersetzung in der Kleingruppen genug Sicherheit erlangt, um selbstbewusst im Plenum aufzutreten. Die Schüler standen hinter „ihrem“ Verfahren, aber sie fanden es auch spannend, wie die anderen Sortierprinzipien funktionierten und welche Lösungsmöglichkeiten noch vorhanden waren.

Die Darstellung der Gewinnklassenauswertung verlief nicht so gut, reichte aber aus, damit alle Schüler einen Einblick in den Lösungsansatz erhielten. Sehr negativ zu bewerten bleibt die Passivität dieser beiden Gruppen während der Diskussion um die Sortierprinzipien. Es wäre besser gewesen, diese leistungsschwächeren Schüler ebenso in die Sortierthematik zu involvieren. Dies hätte geschehen können, indem man ihnen eine Datenliste (z.B. Musikinterpreten, Namensliste) gibt, die sie aufgrund ihrer Erfahrung per Hand sortieren sollten und anschließend ihre Vorgehensweise erläutern. Dadurch wäre zudem die Effizienz informatischer Verfahren stärker herausgestellt worden.

Bei der Präsentation und auch bei der Dokumentation fielen auf, dass die Schüler „moderne“ Medien nicht oder nur sehr wenig benutzten. So wurde statt einer PowerPoint-Präsentation, einer Animation, eines Plakates oder Word-Dokuments fast ausschließlich Handgeschriebenes auf Tafel bzw. Blättern dargeboten. Hier werde ich zukünftig explizit im Arbeitsauftrag den Einsatz anderer Medien vorschreiben.

4.2.4 Unterrichtsprozess II (Entwurf, Implementation, Test)

4.2.4.1. Lernziele

Fachliche Lernziele (FL)

Die fachlichen Ziele des Unterrichtsprozess II liegen gemäß dem Spiralprinzip eine Schleife weiter: Aus den erworbenen Kenntnissen zu den Sortierprinzipien bzw. –verfahren sollen die Schüler ihre Lösung in ein C++-Programm implementieren. Dazu muss zuerst mit Hilfe der GUI für jedes Modul (für Testzwecke) ein geeignetes Formular erstellt werden, welches die Tippeingabe, die Ziehung oder Gewinnklassenauswertung simuliert. Dies ist mit den bereits bekannten Komponenten (z.B. Button, Edit, Label) möglich. Die eigentliche Schwierigkeit während dieser Phase liegt aufgrund der Leistungsfähigkeit der Lerngruppe im Codieren mit C++. Aus diesem Grund bin ich Entschluss gekommen, obwohl es sich bei den hier zu behandelnden Sortieralgorithmen bis auf den Quicksort nur um die geschickte Verwendung von bereits im Unterricht behandelten Themen Schleifen (speziell Dreieckstausch) und Arrays handelt, dass die Schüler das betreffende Programm nicht neu

zu schreiben brauchen. Es soll genügen, wenn sie bereits geschriebene Programme bzw. Programmteile aus Internetquellen oder Büchern übernehmen und es in Hinblick auf ihr erstelltes Formular spezifizieren können. So müssen beispielsweise die Dimension des Arrays und die Anzahl der Schleifendurchläufe angepasst werden. Ebenso sind die Deklarationen und die Ein- und Ausgabe zu ändern.[53] Wesentliche fachliche Lernziele bestehen somit darin, dass die Schüler ein Sortierprogramm dem Formular entsprechend (um)schreiben, den Quellcode vollständig nachvollziehen und interpretieren können und ihr Modul mit adäquaten Testdaten auf die richtige Funktionsweise testen. Eine Erweiterung des bisherigen Wissens stellt der Umgang mit rekursiven Funktionen am Beispiel des Quicksort Verfahren dar, welches einen Anknüpfpunkt für die Unterrichtsinhalte nach diesem Projekt bietet. Hierbei könnte man zudem das Prinzip „Teile und Herrsche“ ansprechen.

Ein sehr lernförderliches Ergebnis erwarte ich in der Integration der Module zu einem Gesamtprogramm. Gelingt dieses Vorhaben, so wird der Sinn der Arbeitsteilung deutlich, wenn nicht, wird der Schwachpunkt der Modularisierung aufgezeigt.

Lernziele, die die Selbständigkeit und Selbsttätigkeit fördern (LSS)

Die Aufgabenstellung, ein Sortierprogramm oder ein Programm zur Gewinnklassenauswertung zu entwickeln, bietet den Schüler die Gelegenheit, während des (Um)Schreibens die Bedienerfertigkeiten und den Umgang mit der Entwicklungsumgebung des C++-Builder handlungsorientiert einzuüben. Sie sollen selbständig Informationen vor allem aus dem Internet herauszufiltern und dabei üben, Informationen zu selektieren, systematisieren und eine Lösungsstrategie daraus zu entwickeln.

Ein weiteres Ziel besteht in der verbalen Vorstellung der Gruppenergebnisse bzw. Lösungsstrategien vor einem größeren Zuhörerkreis unter Zuhilfenahme des PCs als Medium. Dabei sollen die Schüler ihre Produkte (Formular und Quellcode), die sie selbsttätig vorbereitet haben, benutzen lernen Dieses methodische Prinzip der Anschauung bietet auch für zukünftige Präsentationssituationen eine Hilfestellung, um sich in aller Ruhe eine Präsentationen vorzubereiten und durchzuführen. Die Schüler sollen durch eine exemplarische Vorgehensweise mittels konkreter Zahlenbeispiele eine Festigung des erlangten Wissens[54] sowie eine sichere Handhabung bei der Darstellung und Verbalisierung von Ergebnissen erlangen.

---

[53] Anm.: Da die Programme, die im Internet zu finden sind, in C oder C++ geschrieben wurden, sind die Befehle für die Ein- und Ausgabe *cin* und *cout* „objektorientiert“ umzuwandeln.

[54] Vgl. Bruner, 1971, S. 21.

Lernziele, die die Selbstreflexion fördern (LS)

Die Schüler sollen in ihrer Kommunikationsfähigkeit gefördert werden, indem sie ihre Gruppenergebnisse präsentieren, auf Fragen antworten und gemeinsam mit der Kleingruppe, Großgruppe oder Klasse reflektieren. Einerseits sollen sie ihre Ergebnisse verbal vertreten, indem sie ihre Meinung begründen, andererseits sollen sie Verbesserungsvorschläge überdenken und Kompromissbereitschaft zeigen. Dabei kann es zu zwiespältigen oder widersprüchlichen Situationen kommen, die die Schüler durchstehen sollen, um ihre Konfliktfähigkeit zu verbessern.

Für die Selbstreflexion ist es förderlich, dass sich die Lernenden mit den Ergebnissen der anderen Gruppen auseinandersetzen, indem sie die Ergebnisse betrachten, die Ausführungen (Logik des Quellcodes) nachvollziehen, Neues (Verwendung von Schleifen und Arrays) mit Bekanntem verknüpfen sowie die neuen Denk- und Handlungsmuster (mehrere mögliche Algorithmen) übernehmen.

4.2.4.2. Planung, Durchführung und Reflexion[55]

Planung

Bei der Planung orientiere ich mich wiederum an die Phasen des Spiralprinzips und versuche diese methodisch durch die Lernschleife im Unterricht umzusetzen. Wie bereits im ersten Unterrichtsprozess werde ich mich nur in Ausnahmefällen in den Arbeitsprozess einklinken, denn die Schüleraktivitäten stehen im Vordergrund des Unterrichts.

Geplanter Verlauf des Unterrichtsprozess II

Phase	Geplanter Verlauf	Sozial-Aktionsform	Medien/ Materialien
Einstieg	Kurzer Rückbezug auf das Arbeitsblatt	-L-Hinweis	-Arbeitsblatt
Erarbeitung	Absprache und ggf. Herstellung eines gemeinsamen Formulars oder eines Testformular (Um)Schreiben des Quellcodes und aneignen notwendiger Programmierkenntnisse Verstehen der Funktionsweise des Programms	-Gruppenarbeit -L berät	-Arbeitsblatt -Internet -Schulbuch -Standardsoftware -C++-Builder

[55] Anm.: Dokumentationen und Programme (Quellcode oder Formular) der Schüler befinden sich im Anhang.

	Vorbereitung der Präsentation Integration der Module		
Präsentation und Reflexion	Jede Gruppe stellt ihr Programm bzw. Modul vor und erklärt die Funktionsweise Schüler antworten auf Fragen zum Quellcode	-S-Vortrag -S-S-Gespräch -S-L-Gespräch -L moderiert	-PC + Beamer -Tafel
Ergebnis-sicherung	Kopien der Programme werden auf dem Server abgelegt	-Gruppenarbeit	-PC / Schulnetz -Sortierprogramme

Zur Überleitung von der Projektphase 1 zur Projektphase 2 werde ich die Schüler kurz auf das Arbeitsblatt verweisen, damit sie bei ihrer weiteren Vorgehensweise die Anforderungen an die Programme beachten.

Nachdem die Sortierprinzipien bekannt sind, erstellen die Schüler in der Erarbeitungsphase einen ersten Entwurf (= 2. Phase des Wasserfallmodells) des geplanten Programms. Dabei greifen sie bei der Entwicklung eines (vorläufigen Test)Formulars auf bereits Bekanntes zurück und finden einen leichten Einstieg. Die einzelnen Gruppen sprechen sich ab, wie das endgültige Formular ungefähr aussehen soll. Bei der Planung des eigentlichen Programmablaufs in der Kleingruppe werden die grundlegenden Prinzipien des Sortierens in angewendeten Zusammenhängen wiederholt.[56] Wünschenswert ist eine Dokumentation mit Hilfe von Nassi-Shneidermann-Diagrammen (Struktogramme), aber eine ausreichende Kommentierung des Quellcodes halte ich auch für akzeptabel, da die Programme von der Länge und den kognitiven Anforderungen her überschaubar sind. Die Schüler entnehmen den Quellcode aus dem Internet und modifizieren diesen dann entsprechend (= 3. Phase des Wasserfallmodells). Diese Vorgehensweise scheint mir aufgrund der Leistungsfähigkeit der Schüler angemessen, denn für sie ist es einfacher, eine Programmstruktur zu verändern, als ein ganz neues Programm zu schreiben. Der Lerneffekt, die Anwendung und Vertiefung der prozessualen Programmierung, bleibt der gleiche. Anschließend überprüfen die Schüler ihr Programm auf Funktionsfähigkeit und Korrektheit (= 4. Phase des Wasserfallmodells). Ist diese gegeben, folgt die Integration der Module zu einem gemeinsamen Programm. Danach bereiten die Gruppen die Präsentation ihrer Ergebnisse vor.

Während der Präsentationsphase stellt jede Gruppe die Funktionsweise ihres Modul vor und erläutert in einzelnen Schritten die Zeilen im Quellcode. Die Erläuterung sollte anschaulich anhand von sechs Zahlen durchgeführt werden. Einerseits wirkt das Prinzip der Anschauung auf die Lerngruppe motivierend, denn sie können besser dem Programmverlauf folgen. Andererseits erleichtert es dem Vortragenden die Struktur konkret

[56] Vgl. Kap. 3.2.

darzustellen, weil er sich an den Zahlen „festhalten“ kann. Bei der Auswahl der vortragenden Schüler werde ich bereits im Vorfeld festlegen, dass es sich nicht um die selben Schüler wie zur Präsentation der Sortierprinzipien handeln darf. Zum jeweiligen Ende der drei Kleingruppen folgt eine kurze Vorstellung des Gesamtprogramms.
Damit alle Schüler dieses Kurses und auch andere Lerngruppen innerhalb der Schule die Programme zur Verfügung gestellt bekommen, werden diese auf dem Schulserver unter dem entsprechenden Verzeichnis abgespeichert.

Durchführung

Nach längerer Besprechung der Hausaufgabe (Übungsblatt), die von vielen Schüler nur unvollständig bearbeitet wurde[57], begann nach einem kurzen Rückblick auf das Arbeitsblatt die Weiterarbeit am Projekt.
Aufgrund der unterschiedlichen Entwicklung während der zweiten Projektphase erfolgt eine getrennte Beschreibung der Erarbeitungsphase der Gruppen A und B. Beginnen möchte ich mit der Gruppe A, in der die Quicksort-Gruppe (A2) die Sortierung der korrekten Eingabezahlen des Tipps und die Bubblesort-Gruppe (A3) die Sortierung von (Zufalls)Zahlen übernahmen. Die drei Gruppen einigten sich auf ein gemeinsames Formular, welches die Gruppe A2 mittels der GUI entwickelte und aufgrund von Wünschen der anderen Gruppen bis zur endgültigen Fassung modifizierte. Danach erstellte dieses Team ihr Programm zum Quicksort-Verfahren, welches unter geringer Beratung meinerseits gut gelang. Den neuen Syntax die Funktionsweise einer Funktion in C++ verstand die Gruppe selbständig. Sie hatte allerdings einige Mühe, den Ablauf der Rekursion nachzuvollziehen, so dass ich etwas helfen musste. Die Beachtung der korrekten Eingabe der Tippzahlen bereitete keine Probleme und die Ausgabe der Fehlermeldung auf dem Formular war benutzerfreundlich gelöst. Die Schüler hatten zusätzlich die Idee, eine Reihe von Testdaten in das Programm zu integrieren, die alle Fehlermeldungen aufzeigen sollten. Die Gruppe, die das Bubblesort-Verfahren im Quellcode formulierte, fand sich schnell zu recht. Zum einen konnten sie auf eine Vielzahl an Beispielen im Internet und Schulbuch zugreifen. Zum anderen war dieses einfache Verfahren sehr leicht für die Schüler nachvollziehbar, da im Unterricht bereits der Dreieckstausch intensiv behandelt wurde. Die Gruppe, die Gewinnklassenauswertung programmierte, hatte zu Beginn die Schwierigkeit, dass sie nicht wie die beiden anderen Gruppen Lösungsstrukturen durch Beispiele aus dem Internet „inspiriert“ wurden. Sie mussten eine eigene Struktur entwickeln. Das Problem lag für sie darin, wie man einen Code schreiben kann, der alle Zahlen des Tipps mit allen

[57] Anm.: Das Verhalten der Schüler bezüglich Hausaufgaben soll in dieser Arbeit aufgrund der untergeordneten Rolle im Projekt nicht thematisiert werden.

Zahlen der Ziehung vergleicht und die Treffer mitzählt. Mit der Hilfe der anderen Gruppen, gelang ihnen eine Lösung. Nachdem alle Programmmodule gefertigt waren, fassten Holger und Andreas diese zu einem Gesamtprogramm zusammen, während sich andere Schüler auf die Präsentation vorbereiteten. Bei der Gruppenarbeit fiel auf, dass sich primär die männlichen Gruppenmitglieder um die Problemlösung und Programmierung kümmerten, während Verena, Pia und Janine eher nur eintippten oder zuschauten.(FL+, LSS+)
Innerhalb der Großgruppe B gestaltete sich die Zusammenarbeit der einzelnen Kleingruppen schwierig, so dass keine Absprache bezüglich eines gemeinsamen Formulars stattfand. Die beiden Gruppen B2 und B3 erstellten jeweils ein eigenes Formular und die Gruppe B1 vernachlässigte die Arbeit (z.T. auch krankheitsbedingt) sehr stark. (LSS-) Sie schaffte es auch nicht eine Lösung zu erstellen und für die Präsentation vorzubereiten. (FL-, LSS-, LS-) Da die beiden anderen Gruppen nur mit ihrer Aufgabe beschäftigt waren, kam die angestrebte Integration der Module zu einem Gesamtprogramm nicht zustande. (FL-)
In der Gruppe B2 beschäftigte sich Pascal zu lange Zeit mit der Gestaltung des Formulars, da ihm ein wirklicher Tippschein vorschwebte. Dies führte dazu, dass er zwar den Quellcode zum Insertsort korrekt schrieb, aber die Gruppe die Überprüfung des konsistenten Tipps nicht leistete. Ein anderes Problem innerhalb der Gruppe bestand darin, dass Pascal vor allem Elisa (Benjamin hatte sich ausgeklinkt) zeitaufwendig versuchte, den Quellcode des Insertsort zu erklären, weil sie diesen präsentieren sollte und sehr unsicher war. (FL+-, LSS+-) In der dritten Gruppe B3 verlief die Zusammenarbeit sehr harmonisch. Schnell schufen sich die Schüler ein einfaches aber ausreichendes Testformular und setzten sich gemeinsam mit der Programmierung des Selectsort und der Erzeugung der benötigten Zufallszahlen auseinander. Während sie die Ziehung der Lottozahlen vom Prinzip her richtig umsetzten, fanden sie ihren (kleinen) Fehler beim Sortierprogramm nicht, so dass dieses bei „ungünstigen“ Zahlenreihenfolgen nicht richtig funktionierte. (FL+, LSS+) Ich wollte den Fehler zusammen mit der Klasse während der Präsentation ermitteln.
Die fünf Präsentationen fanden am 19.12. satt und führten Rebecca und Daniel, Peter und Andreas, Andreas H., Elisa und wiederholt Nicholas, da sich weder Verena noch Pia aus seiner Gruppe bereiterklärten, durch. Alle Vortragenden, bis auf Elisa, projizierten mittels Beamer den Quellcode an ein Whiteboard und gingen diesen mit einem konkreten Zahlenbeispiel zeilenweise vor. Sie schrieben teilweise ausführlich die Zwischenwerte der benutzten Variablen direkt neben den Code, so dass jeder Mitschüler die Funktionsweise nachvollziehen und auch gezielt Fragen stellten konnte. (FL+, LSS+, LS+) Elisa[58], die sehr nervös wirkte, schrieb den Quellcode der Sortierroutine an das Whiteboard, wobei sie sich

[58] Anm.: Elisa ist eine psychisch sehr labile Schülerin.

des öfteren verschrieb und es im PC-Raum unruhig wurde. Dies führte dazu, dass sie noch unsicherer wurde und den Quellcode unverständlich vortrug sowie auf Fragen nicht antwortete. Da sich die Situation emotional zuspitzte, bat ich Pascal, Elisa im Vortrag zu unterstützen. Dadurch fing sie sich wieder und konnte den Vortrag zufriedenstellend beenden. (FL+, LS-)

Reflexion

Obwohl sich nicht alle Schüler aktiv bei der Programmentwicklung beteiligten, stellte ich ein hohes Engagement innerhalb der Gruppen fest. Die Kommunikation zwischen den Schülern und ihre gegenseitige Unterstützung fanden statt. Ausnahmen bildeten hierbei die Gruppen B1 und teilweise auch B2, bei denen ich rückblickend betrachtet zu wenig intervenierte. Die fachlichen Lernziele wurden größtenteils erfüllt und das Wissen um die Sortierprinzipien durch die Sortierverfahren reorganisiert. Besonders gut gelangen die Repräsentation, denn die Vortragenden konnten in ihrer Sprache die Befehle und Ergebnisse innerhalb des Programms nachvollziehbar für ihre Mitschüler erklären. Ebenso wie in der ersten Präsentation erwies sich die Benutzung der Produkte (=Programme) als hilfreiche und notwendige Unterstützung für den Vortrag. Ein methodisches Problem bestand allerdings darin, dass die Schüler während der Präsentationen und Reflexionen keine Kopien des Quellcodes für Notizen hatten. Dadurch konnte ein Vergleich zwischen den Modulen nicht thematisiert werden, sondern immer nur Anmerkungen oder Unklarheiten zum aktuellen Vortrag geäußert werden. Diese bezogen sich oft auf kleinere Fehler in den Programmen wie z.B. überflüssigen Code beim Bubblesort (Gruppe A3), fehlerhafte Randomfunktion und unvollständiger Quellcode (Gruppe B3). Auch wurde das Verständnis der rekursiven Funktion trotz eines guten beispielbezogenen Vortrags nicht verbessert, so dass das Prinzip „Teile und Herrsche“ nicht angesprochen werden konnte.

Während des Unterrichtsprozesses fiel auf, dass die meisten Gruppen die Dokumentation ihrer Arbeitsvorgänge nicht fortführten. Sicherlich lag dies daran, dass die Dokumenteure sich programmierbezogen nicht so gut auskannten, um detailliert zu beschreiben, was aktuell programmiert wurde. Einerseits brauchten sie Zeit zum Nachvollziehen, andererseits legitimierte für sie das entstandene Programm die Arbeit der Gruppe. Diese Auffassung sowie das Verhalten der Gruppe bzw. des Einzelnen während der Präsentationen werde ich in der nächsten Stunde aufgreifen.

5. Evaluation des Unterrichtsprojektes

Anhand des durchgeführten Unterrichtsprojektes konnten sowohl die zu Beginn beschriebenen fachwissenschaftlichen als auch die didaktisch-methodischen Orientierungen im wesentlichen umgesetzt werden. Auch wenn nicht alle Schüler die gewünschten Leitziele erreicht haben, so durchlebten doch gut zwei Drittel[59] der Beteiligten die für die intellektuelle Entwicklung förderlichen Darstellungsformen nach Bruner:[60] Handlungsmäßig (enaktiv) innerhalb der Gruppe (und anhand des Übungsblattes), bildhaft (ikonisch) mittels der Präsentationen der Sortierprinzipien und symbolisch durch den Quellcode in C++. Die Projektgruppen bewiesen Ausdauer, sich mit einem Thema auf unterschiedlichen Ebenen auseinander zusetzen und sich selbständig und selbsttätig neue Lerninhalte anzueignen. Durch die (fachlichen) Reflexionsprozesse während oder nach den Vorträgen zeigten auch schwächere Schüler, dass sie Potentiale im Hinblick auf Kreativität und Logik besitzen. Die Schüler zeigten sich motiviert, weil sie einerseits im Sinne der Handlungsorientierung eigene Produkte herstellten und andererseits im Rahmen der informatischen Trias ihre persönlichen Fertigkeiten und Kenntnisse in einem ungezwungenen Rahmen verbessern konnten. Für das Vertrauensverhältnis innerhalb der Gruppe, die aktive Auseinandersetzung mit den gestellten Aufgaben und den Mut zu Fehlern waren hierbei die geringe Anzahl der Schüler und das angemessene Anforderungsniveau entscheidend. Das funktionsfähige Programm, welches durch die Integration der Module entstand, war ein echtes „Highlight“ und dürfte genug Ansporn für weitere Projektarbeiten im DV-Unterricht sein. Aufgrund dieser Vorgehensweise (learning by doing) wurde zugleich das ursprüngliche Phasenmodell des Software Engineering vermittelt.

Ob der RLP für den gesamten Ausbildungsprozess einen adäquaten Handlungsrahmen zur Verfügung stellt, kann ich im Rahmen dieser Arbeit eher verneinen. Die Realisierung dieses Projektes ergab sich aufgrund der Umstrukturierung des RLP durch die beteiligten Informatiklehrer. Aus diesem Grund bietet die für das nächste oder übernächste Schuljahr anstehende Novellierung des Lehrplans Datenverarbeitung die Möglichkeit, Unterrichtsprozesse gerade für leistungsschwache Schüler in die in dieser Arbeit beschriebene Richtung zu lenken. Ich hoffe, dass die während des Projektes gesammelten Ideen und Erfahrungen dazu beitragen werden.

---

[59] Anm.: Dieses Ergebnis ist im Vergleich zu den Resultaten aus dem lehrerzentrierten Unterricht wesentlich besser.

[60] Vgl. Bruner, 1971, S. 18ff.

Literaturverzeichnis

Arnold, R., Müller, H.-J., 1993, Handlungsorientierung und ganzheitliches Lernen in der Berufsbildung - 10 Annäherungsversuche, in: Erziehungswissenschaft und Beruf (4), S. 323-333

Baumann, Ruedeger, 1996, Didaktik der Informatik, Stuttgart

Bleßmann, Jörg u.a., 2000, Basiswissen IT-Berufe, Anwendungsentwicklung, Köln

Breier, Norbert, o.J., Empfehlungen für eine Unterrichtseinheit zum Wahlthema „Theoretische Informatik“ in der Jahrgangsstufe 13, Skriptum, Universität Greifswald

Bruner, J.S., 1960, The Process of Education, Cambridge

Bruner, J.S., 1971, Studien zur kognitiven Entwicklung, Stuttgart

Diemer, Wolgang, 1992, EDV, Information, Kommunikation, Paderborn

Freytag, H.-P. u.a., 1992, Der Ausbilder im Betrieb, Teil I, 21. Aufl., Kassel

Dzewas, Jürgen, Cukrowicz, Jutta (Hrsg.), 1991, Informatik, Band 2, Braunschweig

HKM (Hrsg.), 1985, Rahmenlehrpläne für die beruflichen Schulen des Landes Hessen, Berufliches Gymnasium, Schwerpunkt Wirtschaft, Datenverarbeitung, Wiesbaden

HKM (Hrsg.), 1997, Hessisches Schulgesetz, Wiesbaden

HKM (Hrsg.), 2003, VOGO/BG, Wiesbaden

Hübscher, Heinrich u.a., 2001, IT-Kompendium, Braunschweig

Hubwieser, Peter, 2004, Didaktik der Informatik, Berlin u.a.

Kerner, I.O., 1990, Informatik, Berlin

KMK (Hrsg.), 1989, Einheitliche Prüfungsanforderungen in der Abiturprüfung, Datenverarbeitung, Neuwied

Kurbel, Karl, 1990, Programmentwicklung, 5. Aufl., Wiesbaden

Lisop, Ingrid, Huisinga, Richard, 1994, Arbeitsorientierte Exemplarik, Frankfurt/Main

Mäder, A., 1997, 12 Thesen zum handlungsorientierten Unterricht, in: Winklers Flügelstift (2/97), S. 27-38

Magenheim, Johannes, 2003, Informatik Lernlabor – Systemorientierte Didaktik in der Praxis, Skriptum, Universität Paderborn

Meyer, Hilbert, 1994, Unterrichtsmethoden 1, 6. Aufl., Frankfurt a.M.

Modrow, Eckart, 2002, Pragmatischer Konstruktivismus und fundamentale Ideen als Leitlinien der Curriculumentwicklung am Beispiel der theoretischen und technischen Informatik, Dissertation, Universität Halle-Wittenberg

Ottmann, T., Widmayer, P., 1996, Algorithmen und Datenstrukturen, Heidelberg u.a.

Roth, Heinrich, 1965, Pädagogische Psychologie des Lehrens und Lernens, 8. Aufl., Hannover u.a.

Schwill. Andreas, 1996, Vorlesungen der Didaktik der Informatik, Skriptum, Uni Potsdam

Six, Hans-Werner, Winter, Mario, 2004, Software Engineering 1, Kurseinheit 1, Fernuniversität Hagen

Stahlknecht, Peter, 1995, Einführung in die Wirtschaftsinformatik, 7. Auflage, Berlin u.a.

Thomas, Marco, 2003, Didaktische und methodische Gesichtspunkte zur Legitimation und Gestaltung von Lehr-/Lernsequenzen zur Informatik und zum Informatik-unterricht, Skriptum, Universität Potsdam

Wirth, Niklaus, 1983, Algorithmen und Datenstrukturen, Stuttgart

Wolf, 2000, http://www.bildung.hessen.de/abereich/inform/skii/material/delphi/heapsort/

Ehrenwörtliche Erklärung

Ich versichere hiermit ehrenwörtlich, dass ich die vorliegende Arbeit selbständig verfasst und keine andere als die angegebene Literatur verwendet und sämtliche Stellen der Arbeit, die anderen benutzten Werken im Wortlaut oder dem Sinne nach entnommen sind, in jedem einzelnen Falle unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.