Ein genetischer Zugang zum Programmieren mit CGI-Skripten in Python

Jan Schuster

Institut für Didaktik der Mathematik und der Informatik
Goethe-Universität Frankfurt am Main
Senckenberganlage 9
60325 Frankfurt am Main
schuster@math.uni-frankfurt.de

Abstract: In diesem Beitrag wird das genetische Prinzip, das in anderen Fachdidaktiken schon lange eine wichtige Rolle spielt, auf die Informatikdidaktik übertragen. Wie genetischer Informatikunterricht aussehen kann und welche Vorteile dieser bietet, wird in einer Unterrichtseinheit zur Einführung in das Programmieren mit CGI-Skripten in Python gezeigt.

1 Einleitung

Genetisches Unterrichten ist in vielen Didaktiken ein anerkanntes Prinzip. In der Informatikdidaktik wird es allerdings kaum explizit thematisiert. Was das genetische Prinzip für den Informatikunterricht bedeuten kann und wie man es dort umsetzen kann, wird in diesem Beitrag ausgearbeitet.

Im ersten Abschnitt werden zunächst verschiedene Theorien zum Genetischen Prinzip (vorwiegend) aus der Mathematikdidaktik dargestellt, und anschließend wird diese auf die Informatik übertragen. Am Ende steht eine Liste von Kriterien für genetischen Informatikunterricht.

Im zweiten Teil wird am Beispiel der Einführung in das Programmieren mit CGI-Skripten in einem Grundkurs Informatik exemplarisch aufgezeigt, wie diese Prinzipien in die Planung einer Unterrichtseinheit einfließen können.

2 Das genetische Prinzip und seine Bedeutung für die Planung von Informatikunterricht

2.1 Das genetische Prinzip in der Mathematikdidaktik

Das Wort „genetisch“ kommt vom griechischen Wort γίνομαί (ginomai), das „werden, entstehen, geboren werden“ bedeutet. Dazu passend definiert Wittmann für den Mathematikunterricht, dass ein Unterricht genetisch heißt, wenn die Darstellung der Theorie „an natürlichen erkenntnistheoretischen Prozessen der Erschaffung und Anwendung von Mathematik ausgerichtet ist“ ([Wi81], S. 130). Dies bedeutet, dass die Schüler und Schülerinnen[1] Mathematik nicht als fertiges Konzept kennenlernen sollen, sondern den Schöpfungsprozess, der dazu geführt hat, noch einmal nachempfinden bzw. nachvollziehen sollen. Mathematik soll als etwas lebendiges, neu zu Entdeckendes, zu Erfindendes unterrichtet werden.

Der genetische Mathematikunterricht steht im Gegensatz zum deduktiven Stil der Hochschulmathematik, bei dem von Axiomen ausgehend die fertige Mathematik abgeleitet wird. Ein genetischer Unterricht startet beim Vorwissen der Schüler und baut dieses Schritt für Schritt aus.

Wittmann nennt folgende Merkmale, die genetischen Unterricht in der Mathematik (GMU) charakterisieren:

- **(GMU1)** „Anschluss an das Vorverständnis der Adressaten,
- **(GMU2)** Einbettung der Überlegungen in größere ganzheitliche Problemkontexte außerhalb oder innerhalb der Mathematik,
- **(GMU3)** Zulässigkeit einer informellen Einführung von Begriffen aus dem Kontext heraus,
- **(GMU4)** Hinführung zu strengen Überlegungen über intuitive und heuristische Ansätze,
- **(GMU5)** Durchgehende Motivation und Kontinuität,
- **(GMU6)** Während des Voranschreitens allmähliche Erweiterung des Gesichtskreises und entsprechende Standpunktverlagerungen“ ([Wi81], S. 131).

Namhafte Mitbegründer, Förderer und Vertreter der genetischen Methode waren Felix Klein, Otto Toeplitz, Hans Freudenthal, Anna Z. Krygowska, Alexander I. Wittenberg, Martin Wagenschein, Jean Piaget, Jerome S. Bruner; um nur einige zu nennen. Alle teilen die gemeinsame Überzeugung, „dass die Mathematik nur über den Prozess der Mathematisierung richtig verstanden und erlernt werden kann, nicht als Fertigfabrikat“ ([Wi81], S. 131).

[1] Im Sinne der besseren Lesbarkeit wird im Folgenden meist nur noch die männliche Form gebraucht um sowohl Schüler als auch Schülerinnen zu bezeichnen.

Besonders stark prägte Felix Klein das Genetische Prinzip. Seine Vorstellungen für den Mathematikunterricht entwickelte er insbesondere in seinem dreibändigen Werk „Elementarmathematik vom höheren Standpunkt aus“ [Kl24]: „…anschaulich und genetisch, d.h., das ganze Lehrgebäude wird auf Grund bekannter anschaulicher Dinge ganz allmählich von unten aufgebaut; hierin liegt ein scharf ausgeprägter Gegensatz gegen den meist auf Hochschulen üblichen logischen und systematischen Unterrichtsbetrieb.“ (S. 6)

Auch Wittenberg forderte für den Mathematikunterricht eine Wiederentdeckung der Mathematik von Anfang an. Dazu entwickelte er auf Grundlage des genetischen Prinzips seine Themenkreismethode (vgl. [Wi63]). Er fordert eine „Auswahl des Unterrichtsstoffes, die diesen als organische Verknüpfung einiger weniger, bedeutungsvoller, verhältnismäßig umfassender, um eine sehr geringe Zahl einfacher mathematischer Tatsachen angeordneter Themenkreise erscheinen lässt; als einen wohlgestalteten, überzeugenden, überschaubaren gedanklichen Bau“ ([Wi63], S. 141). Dabei sollen insbesondere die letzten Punkte den Schüler überzeugen und nicht den Mathematiker. Für den Unterricht fordert Wittenberg, die „Gegenstände des Unterrichts so zu organisieren, wie sie tatsächlich in einer sachgemäß fortschreitenden Untersuchung ursprünglich hätten erschlossen werden können“ ([Wi63], S. 145). Die Erarbeitung eines Themenkreises soll grundsätzlich in zwei Stufen erfolgen: „allmählich fortschreitende Entfaltung von einem einfachen Ausgangspunkt aus; sodann zusammenfassender Rückblick auf das Erarbeitete, das in der Rückschau als ein durchsichtiger, klar gegliederter, deutlich sachbezogener gedanklicher Bau gesehen werden muss“ ([Wi63], S.147).

Diese Sicht von Mathematik wird auch von ausgewiesenen Experten gestützt: So schreibt R. Thom (zitiert nach [Wi81], S. 130): „In good teaching one introduces new concepts, ideas etc. by using them, one explains their rules of interaction with primitive elements one has assumed to exist, one makes them familiar through handling these rules. It is only later that one will be able to give the abstract definition which allows one to verify the consistency of the theory extended in this way. Mathematics, even in its most elaborate form, has never proceeded otherwise (except perhaps for certain gratuitous generalizations of algebraic theories).”

2.2 Abgrenzung zum Historisch-Genetischen Ansatz

Oft wird bei „genetisch“ an „historisch-genetisch“ gedacht, wie es ursprünglich z.B. von Comenius propagiert wurde: „Am besten also, am leichtesten und am sichersten werden die Dinge so erkannt, wie sie entstanden sind […] Daher möge der Gang der Lehre dem Gang der Tatsachen folgen und das Frühere zuerst, das Spätere nachher bringen“ ([Co61], S. 200). Ein solcher Ansatz möchte mit den Schülern das *tatsächliche* Werden eines Unterrichtsgegenstandes in der Geschichte nachvollziehen. Dies birgt eine Reihe von Schwierigkeiten. So ist es im Schulunterricht normalerweise wenig sinnvoll, jeden historischen Umweg bei der Lösungsfindung eines Problems nachzuvollziehen.

Außerdem vernachlässigt ein historisch-genetischer Lehrgang die eigene Ordnung, die den einzelnen Lerninhalten inne wohnt. Ein solcher Ansatz birgt auch die Gefahr in sich, die Vorerfahrungen der Schüler zu negieren, insbesondere dann, wenn, wie in der Informatik üblich, neue Ansätze und ihre Umsetzungen unter den Schülern schon weit verbreitet sind.

Genetisch ist hier also so zu verstehen, wie dem Zitat von Wittenberg weiter oben zu entnehmen ist, dass die Inhalte des Unterrichts so zu organisieren sind, wie sie ursprünglich hätten erschlossen werden können, orientiert an der Sachlogik des untersuchten Objekts und am Denken der Schüler.

2.3 Genetischer Informatikunterricht

Wie gezeigt wurde, hat das Konzept des genetischen Unterrichts in der Didaktik der Mathematik breite Unterstützung erfahren. In der aktuellen Literatur zur Didaktik der Informatik wird genetischer Unterricht aber bestenfalls erwähnt (z.B. bei [Hu06]). Dies ist bedauerlich, denn Informatikunterricht bietet in besonderer Weise die Chance, das Neue von Anfang an genetisch aufzubauen, weil man viele Gegenstände aktiv konstruieren kann: Man beginnt mit kleinen, einfachen Bausteinen und setzt nachher ein komplexes Produkt zusammen.

Weil man einen aktuellen, modernen Unterricht bieten möchte, ist man andererseits in der Informatik leicht der Versuchung ausgesetzt, neue Konzepte und Techniken den Schülern als Fertigprodukte vorzusetzen. Ein solcher Unterricht zielt darauf, das bereits strukturierte Produkt den Schülern zu vermitteln. Es liegt im Wesen der Informatik zu strukturieren und zu automatisieren und für die Lernenden ist eben dieser Prozess wichtig. Es ist für die Schüler nicht förderlich, gleich das strukturierte und automatisierte Endergebnis präsentiert zu bekommen.

2.3.1 Nicht genetische Ansätze im Informatikunterricht

Informatikunterricht steht immer in der Versuchung, das Neue für besonders gut und wichtig zu halten und deswegen den Schülern die letzten Modetrends zu präsentieren ohne den Weg dorthin mitzugehen. Als z.B. vor 20 Jahren Transputer (parallele Prozessoren) modern waren, wurde vorgeschlagen, diese im Informatikunterricht zu behandeln: Das ist nicht per se verkehrt, aber aus Mangel an Unterrichtszeit verführt ein solches Thema zu nicht genetischem Vorgehen.

Um es klar zu stellen: Genetischer Unterricht soll selbstverständlich aktuelle Themen der Informatik behandeln, insbesondere auch um die Motivation der Schüler aufrecht zu erhalten und um an ihre Vorerfahrungen im alltäglichen Umgang mit Computern anzuknüpfen. Man läuft dabei allerdings schnell Gefahr, diese Themen den Schülern als fertige Wunderwerke zu präsentieren, statt diese mit ihnen selbst zu entwickeln und so das Verständnis zu fördern. Ein Beispiel zur genetischen Behandlung eines aktuellen Themas aus der Informatik wird im nächsten Kapitel vorgestellt.

Auch der hessische Lehrplan Informatik kann ein nicht genetisches Vorgehen provozieren, wenn man sich die detaillierten Vorgaben betrachtet und diese Stück für Stück abarbeitet, statt die dahinter stehende Struktur zu analysieren und den Kurs genetisch zu strukturieren. Dies wird in Kapitel 3 genauer dargestellt.

2.3.2 Charakterisierung von genetischem Informatikunterricht

Für einen zeitgemäßen genetischen Informatikunterricht (GIU) soll folgende Charakterisierung als Arbeitsdefinition verwendet werden:

(GIU1) GIU setzt bei den Vorerfahrungen der Schüler an.

(GIU2) GIU fasst informatische Begriffe als „gewordene" auf und will ihr mögliches „Werden" im Lernprozess nachvollziehen lassen.[2]

(GIU3) Entwicklung von einem einfachen Anfangspunkt aus. Vom Einfachen zum Komplexen, vom Anschaulichen zum Abstrakten.

(GIU4) GIU stellt den Begriff der Entwicklung ins Zentrum der Informatik-Didaktik. Konzepte werden entwickelt, nicht präsentiert. Insbesondere muss erst eine Problemlage von den Schülerinnen und Schülern erkannt werden, und eine Problemlösung gewünscht werden, bevor diese erarbeitet werden kann.

(GIU5) Formalismus sollte minimiert und dort, wo er unverzichtbar ist, entwickelt werden.

(GIU6) Ein „Lernen auf Vorrat" ist zu vermeiden.

(GIU7) Es ist möglich und oft sinnvoll, bestimmte Themen aus dem deduktiven System herausgelöst zu behandeln, ohne vorher alle Grundlagen zu klären. Die Notwendigkeit der Behandlung dieser Grundlagen wird an der entsprechenden Stelle sofort klar und ist dann immer noch und sogar leichter möglich[3].

(GIU8) Anwendungen haben eine hohe Bedeutung. Die Schüler sollen Konzepte nicht nur lernen, sondern von Anfang an praktisch umsetzen.

2.3.3 Informatik im Kontext (IniK)

Genetischer Informatikunterricht (GIU) und Informatik im Kontext (IniK) haben einige Gemeinsamkeiten (Anknüpfen an das Vorwissen der Schüler, weg von der Fachsystematik). Außerdem ist IniK eine gute Unterstützung des genetischen Informatikunterrichts, da sie viele Beispiele liefert, die gut genetisch aufgezogen werden können und ähnliche Ziele verfolgt.

---

[2] Dadurch bekommen die prozessbezogenen Kompetenzen im Sinne der Bildungsstandards der Gesellschaft für Informatik besondere Bedeutung.

[3] So eröffnet sich u.U. eine Möglichkeit zur Binnendifferenzierung.

Genetischer Informatikunterricht bietet mehr als Informatik im Kontext. So geht GIU von einfachen Ausgangspunkten aus und entwickelt daraus komplexe Konzepte, während IniK existierende komplexe Konzepte in ihrem Kontext betrachtet. So lassen sich im genetischen Informatikunterricht auch theoretische Konzepte erschließen, die nicht in das Konzept der IniK passen. Als Beispiel lässt sich das Halteproblem anführen, das in einem genetischen Informatikunterricht über das Debugging beim Programmieren erarbeitet werden könnte.

3 Planung einer Unterrichtseinheit nach dem genetischen Prinzip

Im Rahmen eines Schulversuchs zum genetischen Informatikunterricht wurde in einer Unterrichtseinheit das Programmieren anhand von CGI-Skripten eingeführt. Im Folgenden werden die Planung und die Durchführung dieser Unterrichtsreihe beschrieben. Daran soll gezeigt werden, welche Rolle das genetische Prinzip spielen kann und welchen Vorteil ein genetisches Vorgehen hat.

3.1 Der Schulversuch

Die Lage des Informatikunterrichts gibt Anlass zur Sorge. Viele Lehrer aus verschiedenen Bundesländern berichten, dass die Anwahl der Kurse insbesondere in der Qualifizierungsphase hinter den Erwartungen zurückbleibt. Das gilt sogar an solchen Schulen, die die Informatik in der Sekundarstufe I besonders fördern, z.B. im Rahmen des Wahlpflichtunterrichts. Informatik erscheint also offensichtlich als ein schwieriges Fach, das den Impuls des verbreiteten Interesses an Computern nicht in ausreichendem Maße in Interesse an den Grundlagen der Informatik umsetzen kann. Es scheint, dass gerade der Zeitpunkt der Transformation von der informellen Informatik mit eher spielerisch-experimentellem Ansatz (wie vielerorts in der Sekundarstufe I praktiziert) zur Wissenschaftspropädeutik kritisch ist, dass also die wissenschaftlichen Konzepte der Informatik nicht auf fruchtbaren Boden fallen.

Ein wichtiger Punkt dürfte auch der hohe Aufwand für technische Aspekte des Programmierens sein. Traditionelle Programmiersprachen wie Delphi und Java fordern eine umfangreiche Einarbeitung und überfrachten den Lernenden mit einer Vielzahl von Details. Dies bringt die Gefahr mit sich, dass viele der von Schülern als attraktiv empfundenen Themen (siehe z.B. [RO09]) zu kurz kommen. Der Versuch soll erproben, wie sich die Ideen des genetischen Unterrichts im zeitgemäßen Informatikunterricht umsetzen lassen. Zeitgemäß bedeutet dabei insbesondere, dass die Vorerfahrungen der Schüler und Schülerinnen mit Computeranwendungen, Internet (insbesondere auch dem Web 2.0) und Mobilfunktechniken genutzt werden.

Im Laufe des Versuchs soll Unterrichtsmaterial entwickelt werden, das den genetischen Zugang umsetzt. Dieses Material wird die Chancen des interaktiven und durch Literale unterstützten Arbeitens mit Python intensiv nutzen. Gleichzeitig soll erforscht werden, ob eine – ggf. teilweise – Übertragung des Konzeptes auf weiter verbreitete Programmiersprachen möglich ist.

3.2 Vorerfahrungen der Schüler bzgl. interaktiver Webseiten

Die Schüler haben sich in ihrem bisherigen Informatikunterricht in diesem Kurs mit dem Internet beschäftigt und gelernt, Internetseiten in XHTML und CSS zu erstellen und dabei Textdateien als ein Mittel zur Beschreibung statischer Inhalte (Struktur und Formatierung) kennengelernt.

Internetseiten mit rein statischen Inhalten sind in der Erfahrungswelt der Schüler in Zeiten des Web 2.0 eher selten geworden. Die unter den Schülern verbreiteten Internetseiten (Facebook, YouTube, Google, …) sind interaktiv, können also auf Benutzereingaben reagieren. Dazu muss der Computer eingegebene Daten verarbeiten und der Entwickler nicht mehr nur statischen Inhalt festlegen, sondern Handlungsanweisungen für den Computer in Form von Programmen definieren (programmieren).

Um also an die Vorerfahrungen der Schüler anzuknüpfen, soll das Programmieren anhand von CGI-Skripten eingeführt werden, die Daten aus XHTML-Formularen entgegen nehmen, verarbeiten und eine XHTML-Ausgabe erzeugen[4].

An diesem Vorgehen sieht man auch, dass genetisch nicht unbedingt bedeuten muss, die tatsächliche historische Entstehung nachzuvollziehen, sondern auch bedeuten kann, einen möglichen Weg nachzugehen, wie der zu lernende Begriff / das zu lernende Objekt (historisch) hätte entstehen können. Das Programmieren im ursprünglichen Sinne wurde historisch gesehen natürlich nicht als Erweiterung der Funktionalitäten von Webseiten „erfunden“!

Es muss beachtet werden, dass natürlich auch bei diesem Thema nicht automatisch sichergestellt ist, dass man genetisch vorgeht. So besteht bspw. die Gefahr, den Schülern wichtige Konzepte wie das Client-Server-Prinzip als fertige Gegebenheit zu präsentieren. Daher wurde in der Phase der Einführung des Themas CGI darauf Wert gelegt, dass die Schüler von sich aus erkennen, dass die Ausführung des Skriptes auf dem Server geschehen muss, nachdem sie das Client-Server-Prinzip bereits zu Beginn des Kurshalbjahres im Rahmen eines Experiments ausgiebig erfahren haben[5]. Dies wird insbesondere auch dadurch deutlich und für die Schüler sichtbar, dass die HTML- und die Python-Datei auf den Server hochgeladen werden müssen und die Dateizugriffsrechte entsprechend gesetzt werden müssen.

3.3 Einordnung in den hessischen Lehrplan

Der hessische Lehrplan sieht für das erste Jahr der gymnasialen Oberstufe das Thema Internet und die Einführung in das Programmieren vor. So lernen die Schüler den Aufbau und die Funktionsweise des Internets inklusive des Client-Server-Prinzips kennen und erstellen eigene Internetseiten. Der Lehrplan schreibt zum Thema HTML vor, dass die Schüler auch HTML-Formulare erstellen sollen.

---

[4] Eine detaillierte inhaltliche Darstellung der CGI-Programmierung ist im Rahmen dieses Artikels leider nicht möglich. Deshalb sei hier auf entsprechende Literatur (z.B. [We06], S. 529f) verwiesen.

[5] Dabei haben die Schüler mit ihren Computern schrittweise mithilfe von Netzwerkkabeln und Switches ein Netzwerk aufgebaut und immer mehr Dienste installiert um die Funktionalitäten des Internets abzubilden.

Dies ist ein Thema mit hohem informatischen Bildungswert, allerdings nur, wenn man mit den Formularen auch „etwas anfangen kann“, sprich diese auswertet. Das einfache Verschicken von Emails über die mailto-Funktion ist unbefriedigend und meist aufgrund technischer Schwierigkeiten nicht umsetzbar[6]. Außerdem findet die Auswertung des Formulars dabei durch den Computer im Verborgenen statt, ohne dass der Schüler dies nachvollziehen kann. Es scheint also sinnvoll, gerade hier mit dem Programmieren zu beginnen und zwar anhand einer automatisierten Verarbeitung der Daten aus Formularen. Diese kann clientseitig erfolgen (mittels JavaScript) oder serverseitig über CGI-Skripte, die in verschiedenen Programmiersprachen möglich sind, meist wird allerdings PHP verwendet.

JavaScript und PHP eignen sich durchaus für eine Einführung in das Programmieren, wie zahlreiche ausgearbeitete Unterrichtskonzepte in beiden Sprachen nahelegen. Allerdings sind beide Programmiersprachen auf ihren eigenen sehr eng gefassten Einsatzbereich beschränkt. Außerdem ist die clientseitige Verarbeitung der Daten von geringerer Bedeutung, da die CGI-Programmierung später im Zusammenhang mit dem Zugriff auf Datenbanken zusätzlich an Bedeutung gewinnt und dann wieder aufgegriffen werden kann. Es bietet sich daher an, CGI-Skripte zu programmieren und dazu eine Programmiersprache zu verwenden, die den ganzen Kurs hindurch bis zum Abitur eingesetzt werden kann. PHP ist im CGI-Bereich der Quasi-Standard, lässt sich darüber hinaus allerdings wenig einsetzen. Python bietet beide Möglichkeiten. Es lassen sich leicht CGI-Skripte schreiben (wie dieser Beitrag zeigen will), und auch für den weiteren Oberstufenunterricht ist Python einsetzbar.

Dass Python sich für die Einführung in das Programmieren (insbesondere an der Schule) eignet, zeigt z.B. Ingo Linkweiler in seiner Diplomarbeit [Li02]. Auch das MIT ist seit einiger Zeit in den einführenden Programmierkursen auf Python umgestiegen.

3.4 Inhaltliche Planung der Unterrichtseinheit

Die Unterrichtseinheit baut auf einem vorhergehenden kleinen Unterrichtsprojekt auf, in dem die Schüler eine mehrere Seiten umfassende Website zu einem Reiseziel erstellten, nachdem sie die Grundlagen von XHTML erlernt hatten. Nach der abschließenden Präsentation der Ergebnisse folgte eine Reflexionsrunde, in der die Schüler überlegen sollten, wie sie ihr Projekt gerne noch erweitern würden und dabei unterscheiden, ob sie es noch nicht umgesetzt haben, weil sie selbst es noch lernen müssten oder weil XHTML diese Möglichkeiten generell nicht bietet. Aus dieser Diskussion ergaben sich verschiedene Gesichtspunkte, von denen einer immer wieder genannt wurde: Interaktivität. Hieraus entsteht die nötige Einsicht und Motivation, etwas Neues lernen zu wollen.

[6] Dazu wäre ein installierter und konfigurierter Email-Client auf dem Computer notwendig, was Benutzerkonten auf den Schulcomputern voraussetzt. Dies ist in den meisten Schulen nicht gegeben.

Der Lehrer führt den Schülern eine Webseite vor, die ein Formular enthält, das Daten des Benutzers (Name, Vorname, Alter, Gewicht) abfragt. Nach Absenden des Formulars erhält der Benutzer eine XHTML-Seite, die die eingegebenen Daten aufbereitet (z.B. das Alter in Hundejahre umrechnet und das Gewicht auf dem Mond angibt)[7]. Die Schüler analysierten anschließend die Webseite. Indem sie die Quelltexte betrachteten, konnten sie sehen, dass das Formular mit einer Datei mit der Endung .py verknüpft (die offensichtlich auf dem Server liegt) und dass mit dieser dann durch den Server eine Ausgabe in Form einer HTML-Seite erzeugt wird. Daraus ergibt sich die Frage, wie der Server Texte verarbeiten kann.

Um diese Frage zu beantworten, bekamen die Schüler den Quelltext der XHTML-Seite und des Python-Skripts ausgeteilt und konnten dieses ohne größere Schwierigkeiten sich selbst und den anderen erklären. Die im Quelltext ausgeteilten Dateien wurden den Schülern auch elektronisch zur Verfügung gestellt. Sie ergänzten das Skript um einen weiteren Absatz und hatten so bereits ihre erste Programmiererfahrung mit Python gesammelt.

Das Lernen durch Selbsterklären von ausgearbeiteten Lösungsbeispielen (wie hier) ist u.a. von Renkl ([Re08] Kapitel 4.3.3, S.123ff) und Zöttl ([Zö10]) untersucht worden. Wie Renkl zeigt, ist dieses Konzept vor allem dann erfolgreich, wenn viele Selbsterklärungsaktivitäten vom Schüler gefordert werden. Dies lässt sich in der Informatik besonders leicht umsetzen. In dem hier dargestellten Fall müssen die Schüler sich anhand des Quelltextes z.B. erklären, wie eine Zeichenkette mit einer Zahl multipliziert werden kann. Weitere Selbsterklärungsaktivität wird hervorgerufen, wenn die Schüler das Skript erweitern sollen.

Auf dem bisherigen Unterrichtsverlauf aufbauend können nun alle wesentlichen Themen eines Kurses zur Einführung in das Programmieren anhand der CGI-Skripte behandelt werden, um später darauf aufzubauen. So werden Verzweigungen genutzt, um abhängig von den vom Benutzer eingegebenen Daten spezifische Rückgaben zu machen oder auch Checkboxen abzufragen. Jetzt lässt sich ein Feedbackformular erstellen, das als Erweiterung für das Reiseziel-Projekt eingeführt wird. Außerdem könnte sich dann die Frage stellen, ob es nicht möglich ist, einen Emailverteiler zu verwalten. Die abgesendeten Daten lassen sich leicht in eine Textdatei speichern und dann mit Hilfe einer Schleife zeilenweise auslesen. Um andere Schleifentypen und verschachtelte Schleifen einzusetzen, ließe sich z.B. ein Terminkalender erstellen, der wiederum eine ganze Menge algorithmischer Fragestellungen erlaubt.

[7] Das Beispiel ist dem Buch „Python programming for the absolute beginner“ ([Da06], S. 17 ff) entnommen und für die CGI-Programmierung adaptiert.

4 Fazit

Der vorliegende Artikel zeigt, dass das Genetische Prinzip die Informatikdidaktik entscheidend bereichern kann, weil ein genetischer Unterricht enger am Schüler orientiert ist als Unterricht, der einem streng fachsystematischen Konzept folgt und damit schlussendlich erfolgreicher sein kann. Außerdem bietet der genetische Unterricht viele Anknüpfungspunkte an wichtige didaktisch/methodische Konzepte (Kompetenzorientierung, Binnendifferenzierung,…). Informatik ist besonders geeignet, genetisch unterrichtet zu werden, weil das Strukturieren und Automatisieren wesentliche Grundsätze der Informatik sind und es daher unabdingbar erscheint, die Schüler diese Prozesse beim Erlernen neuen Stoffes mit erleben zu lassen.

Dies wurde am Beispiel der CGI-Programmierung gezeigt, die ein aktuelles und wichtiges Thema der Informatik ist und aus didaktischer Perspektive eine sehr gute Möglichkeit bietet, eine Brücke zu schlagen zwischen dem Medium Text zur Beschreibung von Inhalt zum Medium Text zur Beschreibung von Abläufen; konkret gesprochen vom Gestalten von statischen Webseiten in (X)HTML zum Programmieren, in diesem Fall von CGI-Skripten in Python.

Im Rahmen des Schulversuchs sollen weitere Beispiele für genetischen Informatikunterricht gesammelt werden und auch damit weitere Kriterien für genetischen Informatikunterricht erstellt werden um das Konzept des genetischen Informatikunterrichts weiter auszubauen und weiter zu präzisieren. Über den gesamten Verlauf des Schulversuchs während der drei Jahre an zwei Schulen soll beobachtet werden, wie sich die konsequente genetische Planung z.B. auf die Abwahltendenz, die Leistungen der Schüler, die Motivation, die Selbstwirksamkeitserwartung im Vergleich zu Parallelkursen verändert.

Literaturverzeichnis

[Co61] Comenius, J. A.: Grosse Didaktik. VEB Verl. Volk u. Wissen, 1961.

[Da06] Dawson, M.: Python programming for the absolute beginner. Thomson Course Technology, Boston Mass., 2006.

[Hu06] Humbert, L.: Didaktik der Informatik. Vieweg + Teubner, Wiesbaden 2006.

[Kl24] Klein, F.: Elementarmathematik vom höheren Standpunkte aus, Bd. 1. Berlin-Göttingen-Heidelberg, 1924.

[Li02] Linkweiler, I.: Eignet sich die Skriptsprache Python für schnelle Entwicklungen im Softwareentwicklungsprozess? http://www.ingo-linkweiler.de/diplom/Diplomarbeit.pdf.

[Re08] Renkl, A.: Lehrbuch Pädagogische Psychologie, 1.Aufl., Huber, Bern 2008.

[RO09] Rabel, M.; Oldenburg, R.: Konzepte, Modelle und Projekte im Informatikunterricht - Bewertungen und Erwartungen von Studenten. In (Koerber, B. Hrsg.): Zukunft braucht Herkunft. 25 Jahre INFOS ; INFOS 2009. Ges. für Informatik, Bonn, 2009; S. 146–156.

[We06] Weigend, M.: Objektorientierte Programmierung mit Python. mitp, Bonn, 2006.

[Wi63] Wittenberg, A. I.: Bildung und Mathematik. 2. Aufl., Klett, Stuttgart 1990.

[Wi81] Wittmann, E. Ch: Grundfragen des Mathematikunterrichts. 6. Aufl., Vieweg, Braunschweig 1981.

[Zö10] Zöttl, L.: Modellierungskompetenz fördern mit heuristischen Lösungsbeispielen. Univ., Diss.-München, 2009. Franzbecker, Hildesheim, 2010.