SELECTED TEACHING MATERIALS

William Bricken
June 2001

Three selections from my course teaching materials are described below. Each was selected to illustrate a particular aspect of my teaching style, ranging over intellectual provocation, formal mathematical modeling, and code walk-through.

Selection 1: Modeling Computation

Class: SE502 ***Mathematical Foundations***

Theme: There are no unifying concepts which define "computation". Mathematics is an attitude as well as a field of study.

Duration: three class periods (4 hours)

Materials:

1. Overhead transparencies and lecture (one class period)
2. Chapter 0 exercises: philosophy of mathematics (homework)
3. Collation of student responses to Chapter 0 (class discussion)
4. Final project: a map of discrete mathematics (homework)
5. Collation of final project results (class discussion)

Teaching style:

The overheads are used to stimulate and challenge a student's understanding of their profession. The assignment explores the class's use and understanding of mathematics and mathematical philosophy. At the end of the course, the Final Project assesses how well students have developed an integrated model of the field of discrete mathematics. Emphasis on sharing of beliefs and understandings.

Cultural:
Historical and developmental stories from the theory of computation. Compare and contrast various philosophers of mathematics with student opinions. Stories relating the difficulty of simple mathematical questions.

Comments:
An introductory lecture/discussion to establish to the relevance of discrete mathematics for Computer Science.

Selection 2: Abstract Data Structures (ADS)

Class: SE500 ***Data Structures and Algorithms***

Theme: Data abstraction separates mathematical modeling from implementation details.

Duration: two class periods (three hours) lecture
two class periods lab on ADS

Materials:

1. Lecture notes for mathematical structures and data abstraction.
2. Examples, with code, of several domain theories
3. Extended example of the N-dimensional Boolean hypercube (the 2D square)
4. Three related assignments

Teaching style:
The lecture on data abstraction focuses heavily on formal mathematical structures and how to use them both for modeling and in implementation. The ADS for a square provides an in-depth example of how to think about and construct data abstractions, with emphasis on implicit properties. A student's initial understanding of the course content is assessed in Assignment 1. Assignment 2 provides direct experience in constructing a data abstraction. Assignment 3 measures the student's growth in understanding of data structures.

Cultural:
Visual examples of data abstractions. Discussion of how to discover implicit properties.

Selection 3: LISP Coding and Abstraction Techniques

Class: SE553 ***Artificial Intelligence*** (AI)

Theme: AI is the art of successive abstraction.
LISP is the ideal language for abstract programming.

Duration: two class periods (3 hours) lecture
three periods (4 hours) code walk-through and review
three periods discussion and lecture on abstract programming
three periods discussion of student's LISP programs

Materials:

1. public distribution LISP language system
2. Eight LISP programs emphasizing abstract programming techniques
3. Comments on the LISP language
4. Code distribution #2: Abstraction techniques
5. Program modification exercises, parts I and II (homework)

Teaching style:
LISP programs are used to familiarize students with coding techniques, conceptual issues, and approaches to abstraction. Programming assignments scaled for difficulty as homework.

Cultural:
About 50 pages of distributed LISP code. Code review and debugging. Relation of code to programming concepts.

Comments:
The eight LISP programs cover the majority of AI implementation issues (interactivity, logic programming, search, parsing, automated problem solving, filters, object-oriented approaches, and natural language understanding).

COURSE NOTES FOR MATHEMATICAL FOUNDATIONS

Nearly the entire set of lecture notes for the current Fall 2000 course in Mathematical Foundations SE502 is enclosed in Section VII in order to illustrate several points about teaching content and style. Comments follow the listing of handouts.

Commentary

1. Course Information and Syllabus

Students get this on the first day of class. It explains homework, participation, and evaluation policies. I emphasize that I expect the students to take direct responsibility for their learning as mature and motivated adults. This reduces game playing, shallow learning, knowledge as commodity, and the expectation of spoon-feeding.

2. Final Project

I give the final project out during the first week of class so that the students can prepare for it throughout the course. The emphasis is on integration and synthesis, not on specific rote skill acquisition. This more

difficult requirement assures that the students stay focused on the global aspects of the course. Responses from two different classes follow.

3. Chapter 0 Exercises

The first homework assignment is challenging, while acknowledging each student's own skills and knowledge. Mathematical philosophy allows students to reflect on the deeper meaning of the course and its content. It challenges students' expectations of "right answers", easy content, avoidability of thinking, and brittle learning. I compile the results and show the class where they are in agreement, and where they strongly disagree with each other. The main goal, however, is to deconstruct assumption, thoughtlessness, preconception, and the idea of absolute truth. I demonstrate that truth is a function of belief. This is particularly important for engineers, who have little exposure to humanities or to critical thinking. Importantly, the exercise introduces modern mathematical thinking, which intimately includes subjectivity, uncertainty, and falsibility. Several questions challenge mechanistic thinking. During discussion, the class is exposed directly to issues which require broad thinking and critical analysis. The central question, "Why does mathematics work so well?" is supported by an essay by Hamming, a foundational figure in CS. The compilation of results from two classes follows.

4. Formal Symbol Systems

My first goal is to establish the relevance of mathematics to software engineering. I then teach the nature of formality, supporting the lecture with a handout listing the rules of formal systems, rules which no textbook directly teaches or acknowledges. The students are reminded that computation is ill-defined, that computers are currently accretions of old ideas, that there are many different implementations of computation, and that the level of analysis in computational systems is of critical importance. The central uniformity of all formal definitions of computation is stressed. Mathematical models are critiqued and the patterns of growth of mathematical knowledge are studied.

5. Models of Computation

A quick review of the theory of computation, its strengths and weaknesses, and the history of the idea of computability.

6. Logic and Computation

This is a summary of previous lectures, organized as a roadmap for the construction of mathematical models. I introduce the concepts of complexity in the context of deductive techniques and proof. More theory of computation, emphasizing that the essential concepts in the field are inherently paradoxical.

7. The Age of Mathematical Concepts and Symbols

Placing the subject matter into an historical context.

8. Propositional Logic

A quick review of the simplest mathematical system and the importance of the hardest computational problem.

9. History of Logic

An in-depth presentation of the history of the growth of logic, from Aristotle, through scholasticism, to the breakdown of certainty in the early twentieth century. Slides include reproductions of actual historical documents.

10. Logical Proof

Methods of logical proof: tables and natural deduction, in the context of a difficult study problem. Two large sections are not included; the study of Lakatos and the historical evolution of mathematical theorems, and a section of examples of visual, spatial proof.

11. Lattice and Boolean Cube Reasoning

Alternative approaches to logical proof. This section includes an introduction to lattice theory as an organizing principle, and presentation of a dozen different interpretations of Boolean algebra. Boolean cube computation gives students a hands-on experience of logical computation in a spatial format.

12. Deduction Exercises

Exercises exemplifying the application of logical technique to programming.

13. Algebraic Logic

Preparing to join logic and algebra.

14. Predicate Calculus

For organizational completeness, prior to moving on to complex objects (i.e. relations and functions).

15. Boundary Logic

A cultural lecture on a very different spatial approach to proof. Demonstration of the power of boundary techniques in Boolean minimization.

Previous problems in other notations are revisited. This lecture is on my research area. It ends with an impressive physical and visual manipulation (using blocks and fruit) that is morphic to deductive proof systems.

16. Induction and Recursion

The turning point of the course. Mathematical objects have been introduced, along with proof systems. Now everything is phrased in terms of induction. The goal is to demonstrate that all computational objects and problems can be defined by recursive relations. The first step toward deeper mathematical thinking.

17. Sets

A quick review of finite set theory, focusing on morphisms with Boolean algebra. More exercises in recursive definition.

18. Relations

I assume students are familiar with the basic concepts of discrete math. Here the goal is to broaden their understanding by organizing the basic ideas of relation theory in a mathematical framework, and then presenting a dozen different ways you can think about relations. Relational structure and tools are presented.

19. Functions

Functions are presented as a subset of relations. The distinctions between programming with relations and functions are discussed. Then a dozen different ways to think about functions. Mappings are introduced.

20. Algebraic Systems

Moving onto a deeper level of abstraction, the material on logic and functions is cast in the terms of modern algebra (group theory).

21. Exotic Number Systems

Another cultural lecture, again from my research. Although this handout is large, we only cover the basic ideas and set an historical context. The goal is to shock students in a subject they believe they know intimately (integers). Emphasis on the evolution of mathematical concepts. I introduce a new imaginary number here, as well as some substantively different ways of conceiving of and computing with integers.

22. Graphs

I close with the most important mathematical structure for Computer Science. Graph concepts, operations, and algorithms. This is an advanced organizer for the following course in data structures and algorithms. The class ends with students sharing their final assignment papers.

Commentary

These materials and a short discussion of educational goals and approaches have been included to address concerns expressed during my four year review. Most of my foundational classes are similar in style. Advanced elective courses tend more toward discussion and group project work. To summarize the style:

1) Strong emphasis on exposure rather than skill practice.
2) Exposure to a huge amount of material.
3) Deep cultural and historical embedding of the knowledge.
4) Continual challenge to students certainty of knowledge.
5) Exposure to many completely new ideas.
6) Unrelenting sophistication of material
("Welcome to graduate school.").
7) Deconstruction of assumptions and preconceptions.
8) Continual exposure to multiple perspectives and interpretations.
9) Integrated and cross-connected lecture material.
10) Sophisticated teaching strategies.
11) Always entertaining (attention before understanding).
12) Intolerance for shallow, brittle, and rote learning.
13) Disempowering the comfort of stable knowledge.
14) Individual customization of curricula.
15) Tolerance for diversity of opinion.
16) An attempt to leave every student challenged and thinking
after each class.
17) Use of confusion as a technique to sharpen sensation
and attention.

The ultimate goal is long-term learning. I expect students to recall the content of the class after five years.

Several students have difficulty with this approach. I resist categorization, however:

- Students who expect to be hand-feed, who think knowledge is a transferable commodity, who want not to think too hard, and who do not continually question their knowledge are rudely surprised.

- Some students have been required to take this course, others are waived. Those who are resentful of the requirement sometimes criticize the course rather than the requirement.

- Students rigidly expecting a conventional structure in the classroom often get confused. Few leave confused. Generally these students cannot differentiate between alternative structure and "lack of structure".

- Students narrowly focused on job skills are basically in the wrong program. (They should be in one of the hundred institutions in Seattle which teach computer skills.)