Appears in *Proceedings of the 12th International Conference on Technology and Education, 1995, 247-249.*

# A NEW APPROACH TO A UNIVERSITY "COMPUTER LITERACY" COURSE

Ronald Baecker

SOFTWARE — DESIGN, CREATION, USE, AND IMPACT is a new experimental first-year university course designed to demystify computers by empowering students to understand fundamental software concepts and to create simple programs that process text, draw pictures, produce sounds, and interact with their users in novel and imaginative ways.

Software is created in LCSI Microworlds, an environment in which students can easily build and test simple computer programs and interactive applications. A toolkit is planned to aid in this process by enabling them to watch and listen to the operation of their programs. Students are encouraged to enhance their understanding of programs and computer science concepts through collaborative work.

No previous experience with computers or programming is required, nor is an ability with mathematics. The course is designed for students who are able to explore and articulate concepts and express themselves in English. Female student are particularly encouraged to apply. The class is limited to 20 students.

The course also examines in depth significant applications of computers and methods and tools for software design. Understanding is enriched through the study of exemplary application systems built in Microworlds. Significant time is devoted to an analysis of the social impact of computers, including issues such as jobs, control, privacy, and unreliable computers.

Students also work individually or in teams during the second semester on their own projects, which they report on both orally and in writing. Projects deal with a use of Microworlds for a task that is meaningful to the student, an in-depth review of an area of computer application or social impact, or the development of new materials for future versions of this course.

## COURSE SYLLABUS

There are five somewhat overlapping streams of course content — Creation, Design, Use, Impact, and Projects.

Creation — Programming and Software Concepts comprises roughly 10 of the 40 lectures in the course. Here we introduce basic principles of programming in LOGO, with emphasis on the variety of data types with which we can transact in LOGO, procedures, recursion, names and things, and buttons, sliders, and text boxes for building interactive applications in Microworlds. We talk about algorithms, feedback and control, bugs and debugging, and complexity. Major examples include Karel the Robot, the game of Nim, and the Towers of Hanoi.

Department of Computer Science
University of Toronto
10 Kings College Road
Toronto Ontario M5S 1A1 Canada

(416) 978-6983 (phone)
(416) 978-5184 (fax)
rmb@dgp.toronto.edu

Design — Methods and Tools comprises roughly 8 of the 40 lectures in the course. Here we introduce key ideas from system and interface design, stressing user-centred, iterative design and the use of prototypes. We talk of software engineering a little. We demonstrate LOGO-like systems based on other programming paradigms, such as Function Machines, *Logo, and Boxer.

Use — Applications comprises roughly 10 of the 40 lectures in the course. Here we introduce the use of computers in education, simulation and modeling, games, communications and collaborative work, graphics, data processing, forecasting, information visualization, and desktop publishing.

Impact — Issues and Implications comprises roughly 12 of the 40 lectures in the course. We speak of computer crime, computers in the political process, unreliable computers, invasion of privacy, the information society, computers for strategic competitive advantage, artificial intelligence, computerization of the workplace, impacts on the nature of work, gender and computing, and computer literacy.

Projects and Explorations are reported on by the students and discussed by the group in roughly 6 classes.

## ACHIEVEMENTS TO DATE

The first instantiation of this course began in September 1994 and will conclude early in April of 1995. We have successfully expanded the course concept into a complete syllabus and delivered it to 17 first-year students, 6 male and 11 female, all of whom intend to major in something other than computer science or math.

For most of the topics listed under Use and Impact, we have developed a simple Microworlds mini-application that illustrates the topic. These range from a computer-based instructional program to a computer game, from a spreadsheet to a word processor, from a password cracker to an example of unreliable software. We have used these examples to build bridges from the world of programming to the worlds of computer applications and impacts.

Our first three assignments involved LOGO programming exercises. Despite attempts to make them lively and interesting, many students found them difficult and did not enjoy them. Our fourth assignment, happily, seems to be capturing their interest and imagination. They have been asked to imagine a specific kind of personal information manager to be used by a client, who could be a student, a friend, or a relative. They must talk to the client to develop notions of needed functionality, sketch a suitable design, and then implement in Microworlds a small prototype of a personal information manager for the client. Some of the more interesting ideas include an information base of Civil War history, a CD manager, a wedding planner, and a holiday organizer**.**

## CHALLENGES

We are faced with many challenges as we seek to advance the frontiers of the “computer literacy” course genre:

1) Computer science instruction is carried out in a hostile environment that we as instructors would never accept for our own learning. Imagine how difficult it would be for us to learn and use computers as tools for thought and creativity if

our only access was in noisy laboratories with bad air for limited periods of time that often occurred when we were exhausted, and that help were never available when we really needed it.

2) The sixteen students represent an incredible diversity in ability, background, and interests.  How could we deal with 50, or with 400?

3) Despite having access to much of Microworlds source, courtesy of LCSI, modifying the source to construct the visualization toolkit has been difficult and what has been created has been of little use in this year’s class.  For illustrations of what we seek, see Baecker (1981), DiGiano and Baecker (1992), and Price, Baecker, and Small (1993).

4) It has not been obvious how to use collaboration to assist in the students’ learning process, other than to sanction and encourage it.  A study of how students collaborated in this class is now underway.

5) Although we were successful in building Microworlds mini-applications for most of the Use and Impact topics, explaining them in the context of a lecture which must focus mainly on content and not on sample computer programs has proved to be very difficult.  Our goal for next year is to demonstrate the mini-application, but only try to explain one key concept of each implementation making use of a visualization that highlights that concept.  For example, in explaining a bouncing ball simulation we will focus on how the “inner loop” that moves the ball is computing Newton’s Laws of Motion.

6) The course is already packed with material, but we need to include something about hardware, not the way it is often done, with 2-4 early lectures, but with occasional excursions to the domain of hardware throughout the year.

7) A common criticism of LOGO is that is not a real-world language, and students want to learn something they can use.  How, short of teaching Visual Basic, can we achieve this and also preserve the intellectual integrity of the approach?

8) Can variants of this approach be used at other levels, for example, adult education, professional development, community college, or high school?

9) Finally, what is “computer literacy” anyway?  Is it the ability to buy and own a PC?  Or the ability to use a word processor and a data base system?  Or the ability to write simple programs or at least spreadsheet macros?  Or the ability to read and understand what’s happening with computers, and how they are changing the world?  Or the ability to not be intimated by cybercrud?  Or all of these?

## REFERENCES

Baecker, R. (1981), *Sorting Out Sorting*. 30 minute computer-animated film, distributed by Morgan Kaufmann Publishers.

DiGiano, C. & Baecker, R. (1992).  Program Auralization: Sound Enhancements to the Programming Environment, *Proc. Graphics Interface '92*, Morgan Kaufmann, 44-52.

Price, B., Baecker, R., and Small, I. (1993).  A Principled Taxonomy of Software Visualization, *Journal of Visual Languages and Computing*, 4(3), Sept. '93, 211-266.