New Technologies and Innovations in Agricultural Economics Instruction

edited by

David L. Debertin
Mary A. Marchant
Stephan J. Goetz

August, 1992

Proceedings of a Conference and Workshop hosted at the University of Kentucky, February, 1992. This conference was co-sponsored by the Southern Agricultural Economics Association and endorsed by the American Agricultural Economics Association.

The Need for Better Software for Economics Principles: Issues in Implementation

Gerald C. Nelson and Wesley D. Seitz*

Introduction

A recent Journal of Economic Education article made suggestions about features needed for high quality software for economics principles. This paper reports on the development of software that implements many of the suggestions for a principles of agricultural economics course. Key elements include interactive graphs, individualized homework, and integration with other parts of the class.

The Need for "Better Software for Economics Principles": Issues in Implementation[1]

In a recent paper in the Journal of Economic Education, Walbert (1989) commented on current software packages for economics principles textbooks.

> ".. a majority of software packages that accompany principles textbooks today ... are pedagogically naive and mundane. The programs are often little more than electronic versions of the study guide, and the students role is essentially reduced to turning electronic pages." p282

In 1988, we came to a similar conclusion as we began to revise an introductory agricultural economics course and decided to develop our own software, called the AECONIntro™ software[2]. An important concern in the course reorganization was to address four learning styles (see Gregorc 1979 and Helgesen 1986). Some students learn best by listening to a lecture and taking notes. Other students need to read and reflect on the material. A third set of students learn best by discussing the material with other students and the instructor. Finally, some students learn best when they can work on problems that demonstrate the material. The AECONIntro software was developed to provide an interactive learning environment helpful to students who learn best with the final learning style.

Walbert also made 14 suggestions for design of high-quality software to teach the principles of economics. We group these into five categories:

- student involvement with the software,
- immediate feedback,
- increasing level of difficulty,
- high quality graphical models, and
- integration with textbook or lectures.

We also add a sixth category:

- ease of use in a variety of computing environments.

The goal of this article is to discuss the design and development issues faced in developing a particular piece of software and to indicate that it is possible to achieve many of Walbert's goals. In the sections below, we discuss briefly some issues in software design, describe the software we developed, indicate how it successfully meets the six categories described above, and report on how some of the design issues were addressed.

Software Design Issues

In addition to the criteria for assessing a software product suggested by Walbert, it is essential for software developers to consider several "environment" issues as well. If the software is not targeted to fit the environment of

*The authors are respectively Associate Professor and Professor, University of Illinois.

the student-user, it will fail regardless of its inherent quality. Our experience suggests the following six are especially important.

1. What is the level of student computer expertise? In a principles course, many of the students have had little experience and much attention must be devoted to designing the software for a beginner. It is essential to test the software with users having a variety of backgrounds so that no "surprises" occur.

2. What is the level of student subject matter expertise? In a principles course, it is common to have students with weak backgrounds in math and no economics. The software should explicitly develop all of the concepts needed and assume no background knowledge. At the same time, it should provide the option for advanced students to skip over material they already know.

3. What degree of interaction possibilities should be included, what form should they take, and how much guidance should be provided? Both too much freedom without guidance as well as too little freedom and too much guidance can reduce the learning effectiveness of the software.

4. What are the computing resources available to the student? Student computing facilities range widely in sophistication. Students in a principles course are likely to have access to equipment with limited capabilities. Certainly, the more sophisticated the hardware demands of the software, the smaller the audience for the software.

5. Should the software be designed with a single textbook in mind or should it be textbook-independent? The advantage of associating software with a single textbook is that the text and software can be coordinated in terms of order, style, and level of presentation. Thus the software can be fully integrated into the instructional approach used in the course. The disadvantage of associating with a particular text is that the usefulness with other texts is limited and the audience for the software reduced.

6. What are the resources available for development; both software and human? Even though authoring systems have made it much easier to develop computer-aided instructional material, software development is still an expensive process in time and money. Both subject matter and software expertise are needed and it is desirable if the development team includes expertise in educational theory. Ideally, each person on the development team should be familiar with economic concepts, software development, and educational theory.

Software Overview

The software we developed consists of nine HyperCard stacks[3] and a seven page manual. Each of the stacks has an appearance similar to an interactive textbook; it has a table of contents, a series of pages (cards in HyperCard), and a glossary. Students require one to three hours to complete each stack. The most common card contains an interactive graph. All graph cards are divided into the graph space (typically the upper left hand corner) and the dialogue space (usually on the right hand side of the screen, to the left of the control panel (See, for example, Figure 1).

The software was developed with the presumption that few of the users have computer experience. The first stack introduces students to the Macintosh and the operations special to the software. The next three stacks present basic microeconomic concepts including utility maximization and demand curves, product curves, profit maximization and supply curves, and trade. The fifth stack introduces US agricultural policy instruments, and puts the student in the role of a Midwestern corn farmer who must decide whether to participate in the farm programs. The remaining three stacks treat various macroeconomic topics including the production possibilities frontier, the circular flow of goods and services, and the roles of fiscal and monetary policy in determining equilibrium GNP. Students can access each stack directly or through a central switching stack.

Student Involvement with the Software

One of the criteria identified by Walbert was the quality of student involvement with the software. As noted above, much of the earlier economics software did little beyond presenting

questions and answering them on the next screen. In the AECONIntro software, several more sophisticated interaction models are used. The particular approach depends on the concept being presented. In one use of the interactive graph, sometimes referred to as a "movie", the software controls both the presentation of the graphical concept, and the description of the underlying economic principle in the dialogue space. The student controls the pace of presentation with mouse clicks, and can repeat the movie as often as desired.

In a second version, the student has some control over how the graph is drawn, typically by performing some action in the dialogue space. In Figure 1, the student must decide which curve or curves need to be drawn in order to answer the question in the upper part of the dialogue space. In Figure 2, the student chooses levels of monetary and fiscal policy variables and the software generates the appropriate graph. Depending upon the outcome, the student revises the choice of variable levels and redraws the graph. A third version of the interactive graph (not shown) allows the student to interact directly with the software in the graph space. For example, a student can click on two points on a graph to construct a linear demand or supply curve from which the arc price elasticity is automatically calculated.

Immediate Feedback and Increasing Difficulty

The second class of issues stressed by Walbert is the need for immediate feedback. There are two elements of feedback in software design. First, the software should report its progress in implementing actions initiated by the student. For example, when a student answers a question, the software should indicate whether the answer was correct, and if not, either reveal the correct answer or allow the student to try again.

In terms of the subject matter, every query by the student should be provided a correct answer or be informed that the answer was correct. Our approach to this type of immediate feedback is best illustrated by a brief description of the homework exercises and selected examples. Each stack contains from one to four homework exercises (Figures 1 to 3 are from homework exercises). Most exercises take less than 15 minutes to complete, although the most difficult can take up to an hour. The use of random numbers allows common homework exercises with unique solutions. If a student repeats an exercise to improve the homework grade (a practice that is encouraged), the same problem with different numbers is presented. All homework exercises indicate the requirements for full credit and update the student score after each answer. The status of an answer is indicated both visually and audibly. When satisfied with the score the student activates a print homework button which requests the student's name, the TA's name, the name of the stack, and the date completed. After the homework is printed, the score and other information are automatically reset to zero or blanked out.

Walbert argues that "One way to push the student to increase his or her level of performance is to vary problem difficulty in the drills." (p286) Two approaches in the AECONIntro software address this concern. First, increasingly complex variations of some concepts are presented. Second, new questions are more challenging than previous questions in that they assume a higher level of comprehension. The weak student is encouraged to review and repeat exercises until competency is obtained. The strong student is not forced to repeat exercises on material already understood.

In the design of the homework exercises, three different approaches were taken. In the first, the same problem is presented repeatedly with some variation. An example of this is illustrated in Figure 3. Randomly generated demand and supply curves and a world price line are drawn, and a question related to the graph is posed. Correct answers are identified visually (the score is augmented) and audibly (a "pleasant" sound). An incorrect answer generates an unpleasant sound, a reduction in the percent correct, and the correct answer is put after the question. In this first approach the exercise is repeated until a satisfactory homework score is achieved.[4] The questions do not change in difficulty, but if the student understands the

concepts previously presented, the exercise can be completed in five to ten minutes.

In the second approach, of which Figure 1 is an example, a series of questions is asked, each with only one correct answer. Each question is more difficult than the previous one. As in all homework exercises, an incorrect answer is indicated visually and audibly. A correct answer is rewarded by an increase in the score and the next question.

In the final approach to homework exercises, there are several ways to arrive at the correct answer (in Figure 2, equilibrium GNP equal to full employment GNP, with no budget deficit as a secondary goal). A student changes one or more parameters, observes the outcome, and then modifies the parameters again, until a satisfactory outcome is achieved. In this role-playing approach the problems do not increase in difficulty, although in the exercise in Figure 2, 80 percent credit is given for finding a set of policies which generates full employment. Full credit is given if full employment is reached with no budget deficit.

High Quality Graphical Models

Walbert argues that since "Economic instruction at the principles level makes extensive use of graphical models, ... software must use high-quality graphics." (p287) This seemingly innocuous statement presents substantial programming difficulties. Three elements of "high-quality graphics" must be dealt with - visual appearance, degree of interactivity, and speed. The most challenging problems arise in the presentation of the production and cost curves. In order to be useful in an interactive mode, the graphs must be based on mathematical functions whereas graphs in textbooks are drawn to emphasize certain concepts. Furthermore in a text, smooth curves are drawn, but on a computer, linear approximations must be used. There is a tradeoff between number of points (and therefore smoothness of curve) and speed of execution. Figure 1 demonstrates the approach of the AECONIntro software to drawing cost curves. The initial parts of the curves are drawn with large linear segments to reduce computation time; the later parts with short line segments to resemble smooth curves more closely.

Integration with Textbook and Lectures

The remaining class of concerns raised by Walbert relates to coordination with the other forms of instruction. A common experience of instructors who use textbooks with accompanying software is that the software is seldom used unless it is integrated into the class. We used two techniques to improve coordination and encourage use. One approach, described above, is to build the homework assignments into the software. The second approach was to design another stack (called the Lecture Hall stack), based on the layout used in the student stacks, for use in the classroom.[5] The instructor manipulates the lecture material in class using a control panel similar to that in the student stacks (compare Figures 1-3 with the Lecture Hall graphics card in Figure 4). The Lecture Hall text card can be used like an overhead transparency, with the significant advantage that changes can be made quickly and precisely.

The Lecture Hall graphics card provides a work space where examples can be preprogrammed or drawn free hand. We incorporated the interactive movies from the student stacks, modified to allow the instructor to step through them during the lecture. For example, instead of using an overhead with all of the tangencies between the indifference map and a shifting budget constraint already completely drawn, the price of a good can be varied and successive tangencies drawn step by step (Figure 4). The integration of the HyperCard stacks with the lectures allows students to watch the instructor manipulate the software, and appreciate its relevance to the course.

Ease of Use in a Multiplicity of Computing Environments

In addition to the concerns identified by Walbert, we observe that students rarely start a class with a common set of computer skills and often want to use the software in a variety of computing environments. To address the issues of student computer skill level and hardware availability, a combination of software and

"classroom" approaches was used. First, the Macintosh computer was chosen because the software tools available in HyperCard were not available on the PC.[6] In addition, there were a large number of Macintosh computers available on our campus. However, because many of these computers were not equipped with advanced hardware, the AECONIntro software was designed to operate in the minimum Macintosh hardware configuration and from a floppy disk. Second, a special stack was designed to introduce the student to the Macintosh interface and to the conventions used in the software. A one-hour session is scheduled for students in the computer laboratory. After a brief explanation about how to insert diskettes, and the basics of the Macintosh desktop, the students explore the introductory stack independently, with TAs available for assistance. Because the Macintosh was designed with ease of use in mind, it is possible to bring even the most inexperienced beginner up to an acceptable level of competence in a brief period. After this one hour session, students generally do not need more formal assistance in use of the hardware and software.

Conclusions

The software described here demonstrates that it is possible to design software to help economics students who learn best by repeated problem solving and by interacting with the material. It exploits the advantages of computers, and especially HyperCard, in its use of rapid calculations in the background to present graphical concepts and in its extensive use of random numbers to provide repetition with variation. It provides the tools, in the form of the classroom stack, to integrate computer-assisted learning with classroom instruction.

In two areas the AECONIntro software deviates somewhat from the suggestions made by Walbert. Although the software is integrated with lectures, it is not yet integrated with a textbook. At this point we are revising an existing textbook for use in our class. To date, we have used a textbook primarily as a source for information and alternate explanations. However, the software was designed so that it can be modified to fit personal preferences by an instructor (or textbook author) with some knowledge of HyperCard. The features inherent in the HyperCard software plus the features we have designed make some degree of customization straightforward. In addition, Walbert's suggestion of exercises that are progressively more difficult is not always implemented, although the spirit of his recommendation is followed in a variety of ways.

We hope that HyperCard applications such as the AECONIntro software meet two needs of the profession. With relatively little effort, instructors of principles courses in economics or agricultural economics departments should be able to use the material already developed. And the addition of new topics is relatively straightforward. More importantly, we hope that this example of the implementation of educational principles into the economics software can serve as a starting point for others, and that this paper will help others to develop even better software to aid in the teaching of economics principles.

References

Gregorc, A. 1979. "Learning/Teaching styles: their nature and effects" in National Association of Secondary School Principals, *Student Learning Styles: Diagnosing and Prescribing Programs*, Reston, VA.

Helgesen, M. 1986. *The relationship of preferred learning styles of teachers and students to students' rating of their teachers*. unpublished PhD dissertation. University of Illinois, Urbana-Champaign. May.

Walbert, M. 1989. "Writing better software for economics principles textbooks", *Journal of Economic Education* (Summer): 281-289.

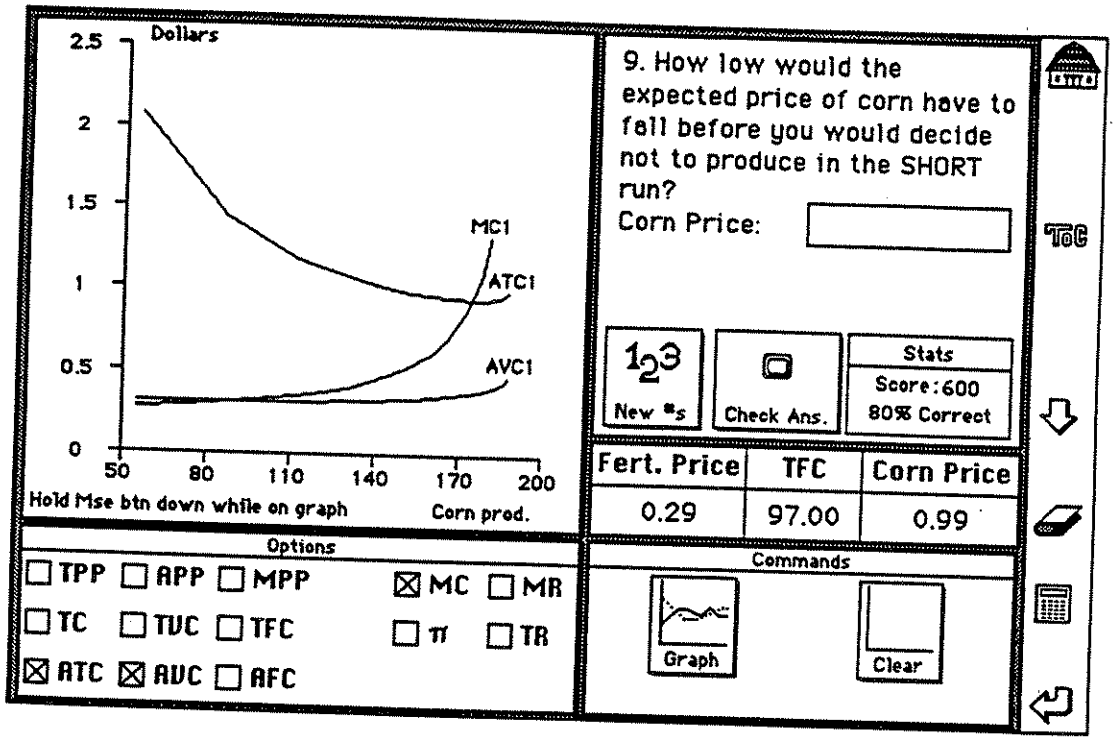

Figure 1. From the Supply Stack

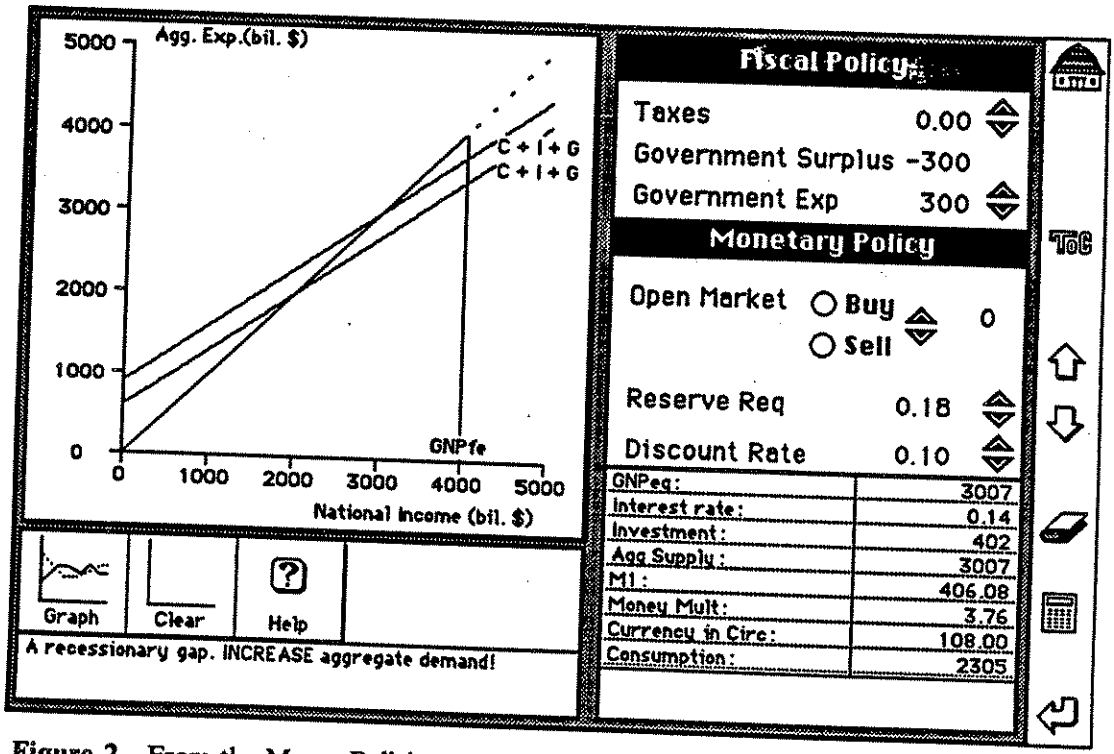

Figure 2. From the Macro Policies Stack

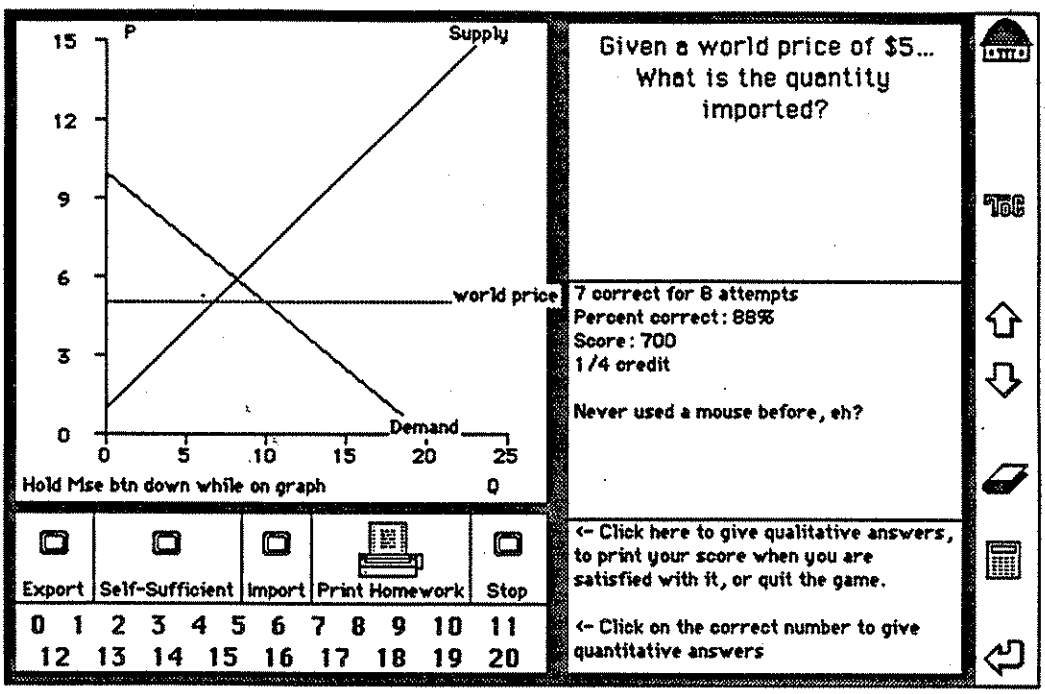

Figure 3. Homework 2 from the Markets Stack

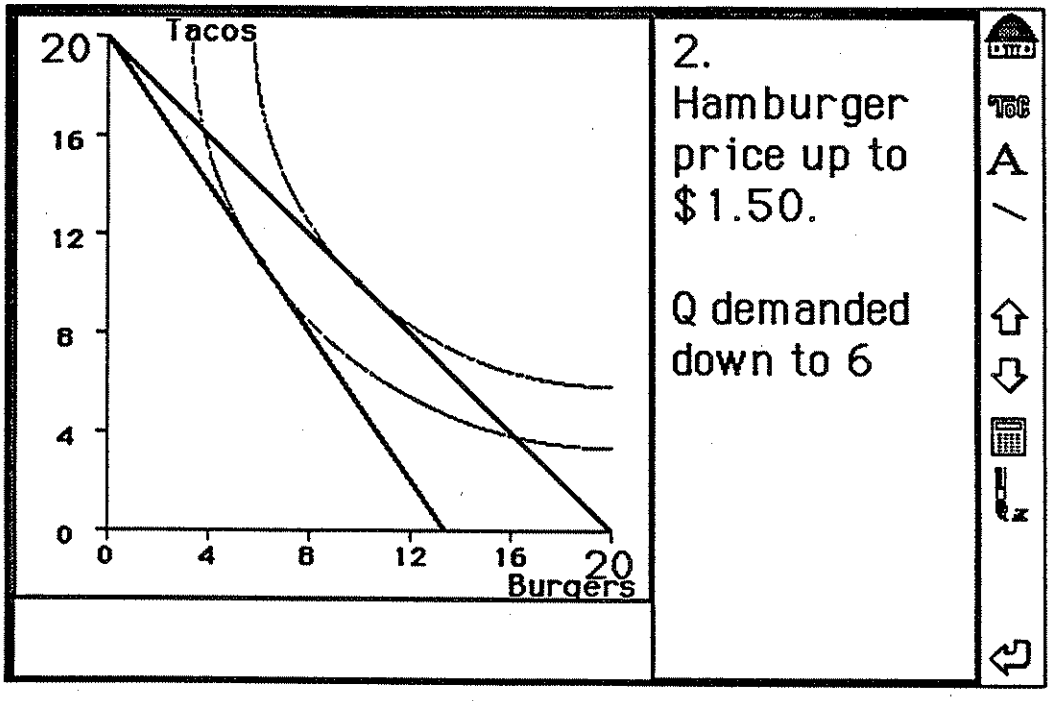

Figure 4. From the Lecture Hall Stack

Endnotes

1. An earlier version of this paper was published in the October 1991 issue of the Technologies in Higher Education (T.H.E.) Journal.

2. Support for the initial development of the AECONIntro software was provided by the University of Illinois, the Apple Corporation, and the Lilly Endowment. The AECONIntro software is available in single copies and with site license arrangements from Intellimation through their academic courseware catalog. It will also be available in both Macintosh and PC Windows Versions with the textbook Introduction To Economics: Resources, Agriculture and Food by Harold Halcrow, Gerald C. Nelson, and Wesley D. Seitz, forthcoming 1994, McGraw-Hill.

3. The HyperCard program comes with every Macintosh. It can be used as a software construction set with elements of object oriented programming and simplified access to standard Macintosh facilities such as buttons, fields, and graphic tools. The metaphor used by HyperCard is a stack of cards (hence the use of the term "stack" to describe a HyperCard document), each of which contains textual, numeric or graphic information.

4. The grading algorithm built in to the software is designed to reduce the likelihood that the student can guess repeatedly until a sufficiently high score is reached.

5. To use this stack, the classroom must be equipped with a Macintosh and projection equipment.

6. In mid-1990 the program ToolBook was released for IBM PCs and compatibles that were capable of running under Windows 3.0. ToolBook is so similar to HyperCard that a program exists to convert HyperCard stacks to ToolBook books. ToolBook also has some features that do not exist even in version 2.0 of HyperCard. However, ToolBook requires a fairly powerful computer in order to run at a "comfortable" speed.