A MODULAR INTRODUCTORY COMPUTER SCIENCE COURSE

Herbert L. Dershem
Hope College
Holland, Michigan 49423

Abstract

The structure of a modular introductory course in Computer Science is described. Two types of modules are offered, lecture and language, over three time periods. Students enrolled for the course complete three lecture and three language modules. Each student chooses modules which match his or her interests and background. In this way the course provides a useful alternative for all students on campus.

Key Words and Phrases: Introductory Course, Modular Course, Service Courses, Languages.

1. Introduction.

The introductory computer science course is a troublesome course because it has to serve the needs of students from widely varying backgrounds and with many different interests. One approach to dealing with this problem is to offer a separate course for each type of student. A second approach, at the other extreme, is offering a single course in which all students must enroll. Many strategies have been employed to strike a balance between the economic infeasibility of the first approach and the pedagogical rigidity of the second.

This paper describes an approach which has been developed and implemented at Hope College from 1978-80 with the support of a grant from the Local Course Improvement Program of the National Science Foundation. Three members of the Computer Science Faculty, Harvey Leland, John Whittle, and the author of this paper developed a set of modules. These modules allow each student to choose a path through the introductory course which is compatible with his or her needs and abilities. The course consists of fifteen modules from which each student chooses six. Three of the six chosen are lecture modules and three are language modules.

 0-89791-036-2/81/0200/0177 $00.75

2. Problems with the Introductory Course.

Before the implementation of the modular introductory course, the following five problems were identified with the course previously offered at Hope College.

Problem 1. Students with varied interests and abilities took the same introductory course. A computer science department consisting of $2\frac{1}{2}$ faculty members and responsible for a growing major program cannot offer multiple introductory courses. About 250 students per year enrolled in the introductory computing course.

Problem 2. There was no appropriate course offered for students desiring a minimal exposure to computers. In spite of student demand, staffing limitations prohibited regular offering of a course for students who did not wish to become proficient in programming, but who merely wanted some exposure to computers.

Problem 3. Students with some prior computing experience had difficulty fitting into the computer science course sequence. An increasing number of students come from high school with prior computing experience. These and transfer students present a special problem because they are usually familiar with much of the material covered in the introductory course, but, for a variety of reasons, are not prepared for upper level courses. Most common among these reasons are that they learned a different language, they did not develop proper programming practices, or they did not gain sufficient exposure to problem solving.

Problem 4. There was a high drop-out rate in the introductory course. Twenty percent of students enrolling for introductory computing did not complete the course. Students usually dropped out of the course because they found it unresponsive to their interests or because too much time was required of them.

Problem 5. Staffing the introductory course was difficult. With growing student demand and a fixed number of faculty, we were faced with the unhappy prospect of larger classes. Matters were complicated further by the fact that all computer

science faculty had duties in some other department, limiting their availability to computer science.

3. The Modular Course.

The course which was implemented to address these problems was a modularized course. The semester is divided into three equal time periods, each about five weeks in length. During each time period, a student is enrolled in one lecture and one language module chosen from among a set offered during that period. Thus a student completes six five-week modules in the course of the semester. These modules are chosen from among fifteen that are offered. The original schedule of module offerings is shown in Table 1.

A brief description of each of these modules is found in Appendix A. These modules were designed by the Hope College Computer Science faculty. Each of the modules meets two hours a week for five weeks. This gives the students a total of four hours of class time per week. Module notes have been prepared for four of the lecture modules and all of the language modules. A textbook has been used for Business Applications, General Applications, and Social Implications.

A student enrolling for the modular course has a choice of a lecture and a language module in each of the three periods. All students enrolled in the course must take Problem Solving during the first time period. This module sets the tone for the entire course and gives the students a common background. All other modules are chosen so as to constitute a course as responsive as possible to each student's needs. Appendix B contains the recommended module selections for some types of students.

This approach has responded to the five problems stated above in the following ways:

Problem 1. Dealing with students with varied interests.

Over the two year development period for this course (1978-1980), the students enrolled completed 19 different paths through the course, meaning that essentially nineteen different courses were taken. The most frequently chosen path, the one recommended for natural science and computer science majors, was taken by 30% of the students. The second most popular path, one recommended for business majors, was taken by 19% of the students enrolled.

Another measurement of the success in dealing with the varied interests of students is the average grade received in the course. During the two years of development, the average course grade was 3.09 based on four points for A, three for B, etc. During the previous two years, the average grade in the introductory course was 2.80. This represents an improvement which is significant at the .01 level.

Problem 2. No offering for students seeking minimal exposure to computers.

An option was provided which allowed students to enroll for lecture modules only and to receive two semester hours of credit rather than the three received for the full course. This option was selected by 85 students over the two year period, thus satisfying their needs for a short, non-programming introduction to computers.

Problem 3. Students with prior computing experience

This course has provided an ideal instrument for handling students with prior computing experience. There is always some path through the course which minimizes duplications of their previous experience. We have successfully required all students to take the introductory course, no matter what their background. This insures a consistent level of maturity and skills in the upper level computer science courses.

Problem 4. High drop rate.

Table 2 indicates the change in drop rate under the new course structure. This phenomenal change in the drop rate can be attributed to two different factors. First, since the students can now choose a path appropriate to their interests, there is less disenchantment with the selection of topics. In addition, those students who find the course too time consuming can change to a less demanding module sequence or drop the language modules and complete the lecture-only course. Thus, options less radical than withdrawal from the course are open to such students.

Problem 5. Limited Availability of Staff.

This problem has been alleviated by this new approach because it gives more flexibility in using faculty members from other departments. Such faculty frequently do not have the time or the expertise to teach the entire course, but they are able to teach modules closely related to their areas of interest. For example, a faculty member from the sociology department has taught the data analysis module. This has the added benefit of exposing faculty in other departments to the topics their students are studying in this course, and giving those same faculty members an opportunity to suggest improvements.

A second related advantage to this course structure is that those computer science faculty members who teach this course every semester, can experience some variety by teaching different modules.

4. Difficulties

Though it has helped to alleviate the above five problems, this modular approach has created two new problems by its very nature. These problems and our approaches to their solutions are presented below.

Problem 6. This multi-path approach complicates the prerequisite definitions for upper level courses

Since students completing this course may have taken any of nineteen module sequences, we must express prerequisites in terms of modules rather

than the course itself. There are, of course, students who decide later they want to take a course whose prerequisite module they did not choose. Such a student is then advised to enroll for that needed module the next time it is offered and complete it without credit.

Problem 7. It is difficult to manage a course which involves so many different instructors and modules.

A comprehensive computerized student information system called SPARS has been developed by Dr. Richard Brockmeier of Hope College. This system has proved to be a valuable tool for this modular course. Using SPARS, mini-registration is held at the beginning of each module period. Class listings and grades are kept in a computerized file for ease of entry and retrieval.

In order to insure proper synchronization of modules, a fairly rigid schedule has been constructed for those modules where coordination is necessary. This insures that lecture and language modules taught by different instructors will be properly coordinated.

5. Evolution of the Course.

As a result of the success of this course over its two year development period, the college has decided to continue this approach. Based on our two years of experience, however, we have made several modifications in the modules offered.

We have restructured the language offerings, eliminating COBOL from the course entirely and moving BASIC to the place previously occupied by COBOL. Our experience showed that COBOL was difficult to teach as a first language, was not friendly to use on our computer system, and did not adapt itself well to the spirit of the course.

The two modules Social Implications and General Applications were found to contain so much overlap that they have been combined into one module entitled Computer Impact. This has allowed us to offer a module called Personal Computing which gives the student experience in using microcomputers and discusses their applications and impact. This new module gives the student who enrolls for the two hour non-language course an opportunity to do some programming. In addition a language module on Microsoft BASIC has been added.

In future years we hope to add Pascal as a language in the course. This would then give the student three language choices in the first two module periods.

6. Conclusions.

Based on extensive data collected during the development of this modular course, it is a significant improvement over the standard course offered previously. This improvement can be observed in better student performance, measured by improved grades; better student attitutde toward the course, measured by lower drop-out rate, a 6% yearly enrollment increase, and improved student evaluations; and better faculty attitude toward the course. Based on this experience, the modular approach appears to be a viable model for the introductory computer science course in a small college.

TABLE 1

Schedule of Module Offerings

	Period 1	Period 2	Period 3
Lecture Modules	Problem Solving	Numerical Techniques	Nonnumerical Techniques
		Business Applications	Social Implications
		General Applications	Data Analysis
Language Modules	FORTRAN I	FORTRAN II	FORTRAN III
	COBOL I	COBOL II	BASIC
		Pre-BASIC	SPSS

TABLE 2

Student Withdrawls from Introductory Computer Science at Hope College, 1976-1980

Academic Year	Mode of Presentation	Enrollment	Withdrawls	Withdrawls as Percent of Enrollment
1976-77	Standard	158	35	22.1
1977-78	Standard	153	35	22.8
1978-79	Modular	147	2	1.4
1979-80	Modular	157	3	1.9

APPENDIX A

Description of Modules

Lecture Modules

1. Problem Solving - This module is taken by all students enrolled in the course. It is designed to introduce any Hope College student to the principles of problem solving using a computer. Topics include introduction to a flowchart language, algorithm construction, and problem solving strategies. This module is a prerequisite for all other lecture modules.

2. Numerical Techniques - Problem solving techniques useful in numerical applications are covered in this module. Topics include the floating point number system, round-off error, finding roots of an equation, numerical integration, least squares, and deterministic simulation. Corequisite: FORTRAN II module.

3. Business Applications - This is a discussion of problem solving techniques used in business data processing and a survey of applications of the computer in business. Topics include file processing, case studies, and information systems. Corequisite: COBOL II module.

4. General Applications - This module is a survey of the uses of computers in all areas of modern life. The roles of simulation, data bases, networks, and computer graphics are emphasized.

5. Nonnumerical Techniques - Sorting, searching, character manipulation, and Monte Carlo simulation are topics treated in this module. Prerequisite: FORTRAN II.

6. Social Implication - This module presents philosophical and social issues which involve computers. These include the effect on employment, electronics funds transfer, privacy, psychological impact, limitations and potentials.

7. Data Analysis - Topics in this module include techniques for entering and storing large amounts of data, analysis and summary of one and two dimensional data, and sampling and hypothesis testing. Corequisite: SPSS module.

Language Modules

1. FORTRAN I - An introduction to FORTRAN is given which includes assignment statements, simplified input/output, control statements, iteration, and subscripted variables. Emphasis in this and all programming modules is on good style.

2. FORTRAN II - Additional features of FORTRAN are introduced including subprograms, formats, and multiple subscripts.

3. FORTRAN III - The use of FORTRAN in the interactive mode is taught in this module. Also introduced are file manipulation and plotting in FORTRAN. Prerequisite: FORTRAN II module.

4. Pre-BASIC - This module is open to those students who have difficulty with FORTRAN I or COBOL I or those enrolled for reduced credit. The elementary featues of BASIC are introduced.

5. BASIC - All features of BASIC are covered in this module. String and matrix manipulation are emphasized. Prerequisite: Pre-BASIC, FORTRAN II, or COBOL II modules.

6. COBOL I - The structure of the COBOL language is taught including all features of the data and procedure divisions.

7. COBOL II - A more complete treatment is given to all language features introduced in COBOL I. This includes file design and disk/tape programming efficiencies. Prerequisite: COBOL I module.

8. SPSS - This module includes features of the Statistical Package for Social Science for data modification, definition, and analysis. Corequisite: Data Analysis module.

APPENDIX B

Typical recommended module sequences

Type of student	Period 1	Period 2	Period 3
Natural Sciences or Computer Science	Problem Solving FORTRAN I	Numerical Tech. FORTRAN II	Nonnumerical Tech. FORTRAN III
Business	Problem Solving COBOL I	Business Appl. COBOL II	Social Implications BASIC
Social Sciences	Problem Solving FORTRAN I	General Appl. FORTRAN II	Data Analysis SPSS
Liberal Arts	Problem Solving FORTRAN I	General Appl. FORTRAN II	Social Implications BASIC
Liberal Arts (2 hr. course/no programming)	Problem Solving	General Appl.	Social Implications
Liberal Arts (2 hr. course/programming)	Problem Solving	Pre-BASIC	BASIC